

分类号: TM863

单位代码: 10335

密 级: 密级

申 请 号: 12345678

浙江大学

硕 士 学 位 论 文



论文题目: 论文题目第一行
论文题目第二行

作者姓名: 作者

指导教师: 导师

专业名称: 计算机科学与技术

所在学院: 计算机学院

论文提交日期: 二〇一五年一月五日

论文题目第一行

论文题目第二行



论文作者签名:

指导教师签名:

论文评阅人 1: 评阅人 教授 浙江大学

评阅人 2: 评阅人 教授 浙江大学

评阅人 3: 评阅人 教授 浙江大学

评阅人 4: 评阅人 教授 浙江大学

评阅人 5: 评阅人 教授 浙江大学

答辩委员会主席: 委 员 教授 浙江大学

委员 1: 委 员 教授 浙江大学

委员 2: 委 员 教授 浙江大学

委员 3: 委 员 教授 浙江大学

委员 4: 委 员 教授 浙江大学

委员 5: 委 员 教授 浙江大学

答辩日期: 2015 年 3 月 10 日

English Title The 1st Line

English Title The 2nd Line



Author's signature:

Signature

Supervisor's signature:

Signature

External Reviewers:	Name	Professional Title	Organization
	Name	Professional Title	Organization
	Name	Professional Title	Organization
	Name	Professional Title	Organization
	Name	Professional Title	Organization

Examining Committee Chairperson:

Name	Professional Title	Organization
------	--------------------	--------------

Examining Committee Members:

Name	Professional Title	Organization
Name	Professional Title	Organization
Name	Professional Title	Organization
Name	Professional Title	Organization
Name	Professional Title	Organization

Date of oral defence: 2015 年 3 月 10 日

浙江大学研究生学位论文独创性声明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得 浙江大学 或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。


学位论文作者签名： 


签字日期： 2015 年 3 月 10 日

学位论文版权使用授权书

本学位论文作者完全了解 浙江大学 有权保留并向国家有关部门或机构送交本论文的复印件和磁盘，允许论文被查阅和借阅。本人授权浙江大学可以将学位论文的全部或部分内容编入有关数据库进行检索和传播，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文。

（保密的学位论文在解密后适用本授权书）

学位论文作者签名： 

导师签名： 

签字日期： 2015 年 3 月 10 日

签字日期： 2015 年 3 月 10 日

勘 误 表

这是一个勘误章节，一般情况下是没有的。

序 言

上一版发布于 2011 年 10 月 26 日，发布之后的近两年来，陆陆续续收到一些邮件问关于使用中的一些问题，我也算基本上做到一一解答。同进也在着手准备根据提到的问题对这一版模版进行一定的修订，增补一些使用中普遍关心的难点问题。因为事务冗杂缠身，加上关于参考文献格式调整部分的内容一直没有时间看明白，这个事情就一直拖下来了。直到前一段断断续续看完了参考文献格式整理部分的帮助资料，搞清楚了它的实现思路原理，才算又着手修订这一版教程。

在这过去的一年多里，接触到了 X_YTeX，对其强大的直接调用系统字体的能力表示赞叹，于是将这个模版切换到了 X_YTeX 的环境下，将文件代码换成了对多语言兼容更好的 UTF-8 代码，但同时保留对 GBK 码的兼容，具体不同之处会在后面的章节中提到。因此新的一版分为 UTF-8 和 GBK 两个版本进行发布，两个版本使用上只有很细微的区别，一般使用过程中可以忽略这个差别。

以下是原来的序言，此处照旧附上。

很早就听说过 L^AT_EX 了，但却一直没有真正学习过，直到今年，需要处理一些大文档，想起了 L^AT_EX。重新翻出 L^AT_EX 的文档，从 CCT 开始，至于为什么是 CCT，因为 C_Tex 提供的那个 C_Tex FAQ 里对中文的第一个例子，就是以 CCT 为例写的。CCT 是中科院的张林波研究员写的，帮助文档都是中文，看起来比较容易，但毕竟是好几年前的版本了，更新也并不是那么及时，而且 CCT 早期版本的字体是点阵字体，边缘很粗糙，虽然不影响打印，但在这个年代还在用着这样的字体，着实不是那么舒服。我又开始了第二个例子，CJK 的尝试，在尝试 CJK 的过程中，无意中看到了 C_Tex 的 ctexart, ctexbook 和 ctexrep 这几个基本模版，这才找到 C_Tex 的门，筒子们不要笑我绕了这么一大圈才摸进了 C_Tex 的门，虽然从开始就使用的是 C_Tex 的发行版。

这里也要说一下，C_Tex 提供的部分帮助文档内容也比较老了，一些操作现在新的软件虽然仍然兼容，但已经不是新版软件推荐的做法了，比如，C_Tex FAQ 里面对于 pdf 文件的生成，依然是先由 latex.exe 生成 dvi 文件，再由 dvi 文件生成 ps 文件，最后再生成 pdf 文件。实际上，现在流行的新版 T_EX 类软件都已经将 pdf_Tex 作为默认引擎，支持直接生成 pdf 文件，而且 dvi、ps 文件的打开速度比 pdf 反而要慢许多。我使用的是 64 位系统，C_Tex 提供的安装包只支持 32 位系统，我单独安装的 MikTeX x64 版使用 C_Tex 模版生成的 dvi 文件使用 dvips 处理时会找不到字体，因为这个问题，我找了很久，最后的结论是：

dvips 可以放弃了，直接使用 dvipdfm 更合适。

后来几天在 \LaTeX 的实践中看不少相关细节，开始对其模版产生了兴趣，在 88 上 \TeX 版把置顶的 ZJUthesis 下了下来，就是写这个模版的基础，数学系模版。下下来后发现这个模板给的例子 pdf 与当前学校使用的 2008 年论文模版差别老大了，从封面到目录，章节格式，都是完全不一样，因此，决定着手做一个与学样提供的 Word 模版比较接近的模版。

在以 2006 年数学系模版为基础进行新模版编写的过程中，学了不少方法，也发现老模版不少过时或者不合适的地方。第一个学到的就是，从模版一开头就发现这个模版是以 ctexbook 这个模版为基础制作的，做到模版完成的时候，发现 88 的 \TeX 版置顶模版已经更新，我以为我白做了，下下来一看，原来这个新的模版不是以 ctexbook 为基础制作的，而是更基础的 $\text{\LaTeX} 2_{\epsilon}$ 对比自己基本完工的模版，才发现 ctexbook 为我省了很多工作量。只是一些修修改改就做到了很接近学校 word 模版的效果。ctexbook 的新版已经直接将 hyperref 包打了进去，2006 年数学系模版对 hyperref 的引用判断部分已经明显过时，在用新版 MikTeX 运行的时候直接报错了。在编写封面的时候，发现 2006 年的模版用了一个五列的表格，可这部分的内容只需要两列就够了，直到我某天下载了中科院的模版后才明白，2006 年版模版是从中科院模版改编而来，中科院模版在封面上名字等内容的排列方式需要采用五列表格。这一部分，我也将其重新编写。

随着时代的推进， \LaTeX 的各种功能包日渐丰富，很多过去只能从 $\text{\LaTeX} 2_{\epsilon}$ 代码写的功能，如今可以通过相应的功能包直接实现，在这个模版中，我使用了几个新的功能包，其中最新的当属刚刚发布的 hyperref 更新包，增加了 hidelinks 命令，可以直接将链接的边框去掉，不用采用将边框颜色设为白色的方式了。

就像 \LaTeX 的版本总是在接近 π 的值一样，这份模版并不是完美的，比如对数学系的定理体系支持不足，留在以后版本再发布或者请有兴趣的爱好者共同修改。编写这一版本的基本目的是没有任何 \LaTeX 基础的同学可以比较轻松地利用它给自己的毕业论文排一个满意的版面，整个模版没有留太多选项，可供修改的选项只有两个：单面双面的选择和链接的颜色的有无。在模版中，我对绝大多数的语句，都做了中文注释，解释其作用，方便有兴趣的同学研究，我也是一个初学者，作出的这份模版，我想，应该比较适合初学者胃口的。

摘 要

这份文档主要介绍了该 \LaTeX 论文模版的使用方法，注意事项，一些使用技巧。

另外，这份文档的源代码修改自 shuwei1204@163.com 的 google code 项目，该项目的下载地址为：<http://code.google.com/p/zjuthesisetex/downloads/list>。

关键词： \LaTeX ，论文模版，ZJU

Abstract

The quick brown fox jump over the lazy dog.

T_EX

Keywords: T_EX

目 录

勘误表	I
序言	II
摘要	IV
Abstract	VI
目录	
插图	XI
表格	XII
缩写、术语表	XIII
第 1 章 简介	1
1.1 为什么	1
1.2 模版简介	3
1.2.1 L ^A T _E X 简介	3
1.2.2 模版内容	4
第 2 章 软件环境及设置	5
2.1 Windows 系统	5
2.1.1 32 位系统	6
2.1.2 64 位系统	7
2.2 linux 系统	8
2.3 Mac 系统	8
2.4 本模版所需要的扩展包	8
2.5 测试运行	8
第 3 章 使用该模版	9
3.1 模版选项	10
3.2 论文封面题名信息	11
3.3 实战：生成自己的论文第一页	12
3.4 完成题名页信息	14
3.5 正文部分各章节的书写	16
3.5.1 勘误页	19

3.5.2 致谢	19
3.5.3 序言	20
3.5.4 摘要	20
3.5.5 英文摘要	20
3.5.6 图片目录	20
3.5.7 表格目录	21
3.5.8 缩写、符号清单、术语表	21
3.5.9 目录	21
3.5.10 正文章节内容	21
3.5.11 参考文献	23
3.5.12 附录	25
3.5.13 索引	25
3.5.14 个人简历	25
3.5.15 发表文章目录	25
3.6 插入图片	26
3.6.1 插入图片的应用进阶	29
3.7 插入表格	30
3.7.1 普通表格的插入	30
3.7.2 把表格换个样式	32
3.7.3 有单元格合并的表格	33
3.7.4 其它高级表格格式及扩展包简介	34
3.8 公式及其编号	34
3.9 索引	35
3.10 参考文献	35
3.11 脚注	35
3.12 重点字符，醒目标示	36
3.13 插入算法伪代码	37
3.14 插入源代码	38
3.15 插入定义、定理等	38
第4章 参考文献格式处理	39
4.1 一般处理方案	39
4.2 专业处理方案	39

4.3 参考资料	40
4.4 L ^A T _E X 参考文献组织思路	40
4.5 如何让 BibTeX 按你的想法干活	41
4.6 关键所在: *.bst 文件	41
4.6.1 bst 文件的结构	42
4.6.2 BibTeX 编程的规则	43
4.6.3 bst 文件的调试技巧	55
4.7 *.mbs 与 *.dbj 文件	57
4.7.1 一般的 bst 文件的生成过程	57
4.7.2 有所不同的注释方式	58
4.7.3 集大成的 *.mbs 文件	59
4.7.4 *.dbj 选项文件	60
第 5 章 一些反馈的问题	62
5.1 关于使用 author year 参考文献引用方式的问题	62
5.2 关于 chapter 居中格式的问题	63
5.3 关于章级目录有时居中有时不居中的解决方案	63
5.4 关于标题两行还写不下的问题	63
5.5 目录层次与子目录分层缩进	64
5.6 关于分章参考文献的用法	65
5.7 第 X 章格式的修改	66
5.8 多个参考文献文中标格式	66
5.9 关于每一章标题头上的空白部分	67
5.10 GBK 与 UTF 版本的问题	67
第 6 章 其他一些使用技巧	68
参考文献	69
附录 A - 贡献者	70
附录 B - 版本更新	71
索引	72
作者简介	73
发表文章目录	74
致谢	75

插图

图 2.1	MiKTeX 升级设置.....	6
图 2.2	选择升级方式，如图中选择一个升级站点	6
图 2.3	选择升级站点，以中科大镜像点为例	7
图 2.4	设置 MiKTeX 软件的 Root 目录	7
图 2.5	刷新 MiKTeX 的目录数据库.....	8
图 3.1	运行 LaTeX	14
图 3.2	跟文字一样宽的图	28
图 3.3	原图一半大小并逆时针旋转 90 度的图	29
图 3.4	matlab 作图例	30

表 格

表 3.1 章节命令 22

表 3.2 表格标题 31

表 3.3 粗细线表格示例 32

表 3.4 复杂表格示例 33

缩写、术语表

缩写、符号清单、术语表

第1章 简介

1.1 为什么

“为什么用 \LaTeX ?”

当把 \LaTeX 介绍给一个人的时候，这是面对的第一个问题。当回答“它很好用时”，又会得到第二个问题：“Word 不好用吗？”答案当然是肯定的，Microsoft Word 是这个世界上目前最流行的，最易用的文字处理软件之一。

但是，我这里要做一个转折了，如果是做过比较大的文档的，几十页以上，有分章，分节，或者还要有页眉页脚页码目录以及封面这些东西，好吧，大部分人做过唯一的这种文档，那就是毕业论文了。给毕业论文排版，几乎是大部分人的梦魇。如果性格里再带一点点追求完美的念头，追求一点点排版的质量，那给自己的毕业论文排版，那一定会留下一生深刻的印象！

页码格式不对，页码编号不对，不同章节标题字体不一致，小标号缩进老对不齐，数字有时候是 Times Roman 字体，有的时候又成了宋体，有的段落前有一块空白，有的段落间距又太小。增加了一张图，它后面图的标号又得重新来编一次，还有图的引用处也不要忘记……增加了一段文字，发现排好的图又跑到下面一页去了。。。。这一页留下了好大一片空白。下一页的表格又被它推成了分布在两页。参考文献引用增加或者删除了一个，那就得找出全文要改动的所有引用处！！什么，你是高级用户，会使用交叉引用，好吧，那有时候打出来的文档突然出现“错误！找不到引用-”这样的字样，你是什么心情？生成目录字体不一样，有的标题起的名太长，把页码挤到下一行去了，一个个改好了，一全更新又完了。如果你的文档里还有一些公式，那又是一场战争了，有的公式字显得比正文大，有的又显得比正文小，或者总是跟写的编号对不齐，不是偏上就是偏下。公式编辑器版本众多，换台电脑就只能看不能改了，公式编辑器还经常提示已过期。blablabla……

千辛万苦，终于搞定了，看起来还算漂亮，到打印店去了，omg，word 版本不对，版白排了，又乱了。用 PDF 吧，不同软件生成的 PDF 总是跟原来的样子有点差距。做完论文，感觉是扒了一层皮。

\LaTeX 解决了这个问题，它实际可以看成是一种写给机器的语言，把格式定义好，再填上内容，它就会按设定好的格式把一份文档生成出来，一般是生成 pdf 文档，整个文档的格式首先是统一的，不会出现不同章节的显示不一样的问题，并且，文档的格式可以是

封装好的，使用的时候不需要理会具体格式，直接可以使用，封闭格式这个功能留给会的人去做，写论文不需要关心太多的格式问题。

如果是往国外投文章，那么 \LaTeX 的应用更广泛，不少国外出版社直接提供其格式文件，只要将文档头上的 `\documentclass{article}` 大括号里的 “article” 换成其相应的格式文件即可，避开了格式调整问题。

话说回来，Word 还是有优势的，所见即所得，上手容易，会鼠标键盘就会用。 \LaTeX 还是需要一些入门时间的，如果单纯会使用这个模版，我想大约需要 3 个小时左右，如果会处理一些常见的程序错误以及做小的格式修改，时间就会比较长了，大约要几天，有人引导的话会快一些。Word 在做小文档，比如就几页的文档上的优势 \LaTeX 是无法与之比拟的，做起来是很快，比如通知啦，传单啦。但 Word 存在的意义应该不是就为了那几张小广告的，几百 M 的体积，几百 RMB 的价格，当然盗版很多。在这种应用上还不如去用一下免费的 WPS，或者 OpenOffice。Word 支持很多特性，支持宏，支持 Visual Basic……，但是，它们又太难了，不是随便一个人就会去有兴趣学它们的，学了很少有机会用到，屠龙之技。

\LaTeX 还有两个优势是 Word 所没有的：

1. 文件体积小

使用 \LaTeX ，所要编辑的文件以 “tex” 为扩展名，如果用到参考文献，可能还需要扩展名为 “bib” 的参考文献数据库文件，此外，还可能有的就是文档中要插入的图片文件了。“tex” 和 “bib” 文件都是纯文本文件，如果愿意，可以用记事本来编辑，按照一定的格式书写，格式也是很简单的，看到了就会使用。而 Word 文件是 Microsoft 自己定义的二进制文件，只有用 Word 软件或者其它兼容的软件打开，因为是 Microsoft 自己的二进制格式，因此，体积会比较大，当然文件集成度很好，只有一个文件。相比较之下 \LaTeX 可能有很多文件，但这个缺陷完全可以通过压缩软件打包来完成。如果打开一个比较大的文档，机器破的话会比较慢，而且容易出一些错误导致 Word 意外关闭，想来各位都遇到过这种情况的吧。文件如果发生了意外关闭，那就有可能被损坏，损坏后就有可能。。。。打不开了，如果没有做一些备份，那就成了“杯具”甚至“餐具”了。相较而言， \LaTeX 的文件小，而且是纯文本文件，即使被损坏，修复起来也比 Word 要容易得多。

2. 使写作更加专注于内容

说实话，这一条在学会使用 \LaTeX 之前，我觉得是扯淡，那时的我觉得，使用 Word 边写边想也一样很快的。但在我学习使用 \LaTeX 的过程中，我逐渐感受到，当只面

对文本，不去想它下一段怎么排，什么样的格式时，思维更加连续，写起来也更快，而且明显感觉到自己进入了一种写作的状态，这种感觉只在以前纸上写作感觉得到，专注于自己想的内容。这一点，只有在学会使用之后，才能去体会得到。

1.2 模版简介

该模版以 CTeX 社区发布的 ctexbook 模版为基础，在数学系 2006 年模版上修改而来，删除了不兼容的旧代码，增加了一些新版本扩展包支持的高级命令，部分功能实现方式与旧版有所不同。

该模版与研究生院网站发布的 2008 年模版排版效果基本相似，可直接使用。

该模版非学校官方模版，该模版可能引起的问题，**模版作者不承担任何责任**，特此声明。

1.2.1 L^AT_EX 简介

L^AT_EX 不是单指一个软件，而是指一类软件，这一类软件都是以一个基本软件 T_EX 为基础，制作成宏包并经 T_EX 的原作者授权后发布。打个比方，T_EX 就是一些方形，三角形，圆形，半圆形的积木块，L^AT_EX 就是另一些人用这些积木块搭成小房子，小桌子，我们再把这些做好的小房子，小桌子摆摆成为我们的积木城市群，就是这样。

为了国际化，T_EX 不读“太可斯”，而按原作者 Knuth, Donald Ervin 的说法，应读为“太 chi”^[1]。

Kunth 是一个计算机与数学家，T_EX 是其在 1977 年看到自己的成果出版时印刷质量甚不满意，于是历时 5 年编写了 T_EX 排版系统，这成为西文排版业的一次重大革命。T_EX 系统于 1982 年正式定型，不再做大的改进，只修正发现的错误。1989 年，T_EX 系统做了迄今为止最大的一次改进：支持多语言^[1]。

T_EX 的设计思想很简单：把一张纸看成一个坐标平面，将该坐标平面上哪一点要出现的内容标记出来，就像这样：坐标 (50, 50)，放置一个五号宋体的字，倾体，不加粗；从坐标 (80, 50) 到坐标 (80, 200)，画一条粗为 2 的线，黑色……一个排好的版面就是这样被一个点一个点描述地画出来，Knuth 设计的 T_EX 指令，就是这些描述指令，然后由计算机解读，并做出最终的图，就是我们看到的排版效果。T_EX 的精度很高，它的最小尺寸是一个叫做 sp 的小单位。可见光的波长近似等于 100 sp，几个 sp 的误差眼睛是看不出来的^[2]。

从上面不难想到，如果只用 $\text{T}_{\text{E}}\text{X}$ 指令进行排版的话，会比较复杂，普通人难以胜任。作为计算机专家的 Knuth 很清楚这一点，他给 $\text{T}_{\text{E}}\text{X}$ 设计了扩展接口。利用这些接口，可以把 $\text{T}_{\text{E}}\text{X}$ 的一系列命令封装起来，做成各种各样的“宏”，这些宏，就被称为 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 。就像上面的比喻一样，用积木做小房子，小桌子的厂家有很多，于是就有了各种各样的 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 版本：MiKTeX, XeTeX, TeXLive, teTeX, fpTeX 等。使用 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 模块，使得排版成为一件容易的事，尤其是使用已制作好的模版写文章，不需要任务基础，知道几条语句就可以排出整齐统一的版面。使用哪个版本的 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 没有关系，都可以使用这个模版生成论文。这个模版制作使用的 LaTeX 版本是 MiKTeX 2.9 版。

1.2.2 模版内容

按照研究生院提供的 2008 版模版，毕业论文一共由封面、题名页、版权声明、勘误表、致谢、序言、摘要、图表目录、术语表、目次、正文、参考文献、符录、索引、简历、文章列表，一共十六部分组成¹。

本模版将图表目录分成了图片目录与表格目录两个部分，因为这样一来清楚，二来 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 原生支持图片与表格分开作目录，作到一起反而费事。其它部分与研究生院模版相同。

在该模版中，任何一个部分都是可自由选择有无，如勘误表，大部分论文是没有这个部分的，不需要某部分，只要将其生成语句注释掉即可，具体将在后面章节中讲解。使用该模版，封面、题名页、版权声明、图片目录，表格目录、目次、参考文献、索引这八个部分不需要论文作者直接参与，只需按要求在文中作出标记， $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 就会替你自动生成这些部分，而且完全不必考虑条目的编号顺序等问题。

¹其实，还有脚注没有算上，脚注是隐含的，在这里 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 将主动替你完成脚注的插入，而无需多作分心。就像这个脚注一样。

第2章 软件环境及设置

该模版是基于 CTeX 社区发行的 ctexbook 模版，因此要使用该模版需要有 CTeX 环境。

与第一版稍有不同的是，由于该版提供了采用 UTF-8 编码的 X_YTeX 引擎的新版本和使用原有采用 GBK 编码的 L^AT_EX 引擎的老版本，因此，如果使用新版本，还需要参考本模版文件包中提供的 ctex-xecjk-winfonds.def 文件将自己系统中的 ctex-xecjk-winfonds.def 文件进行修改，该文件位于 ctex 目录下的 \tex\latex\ctex\fontset 目录下。需要至少指定以下六种字体对应的字体文件：

```
\setCJKfamilyfont{zh song}{SimSun}
\setCJKfamilyfont{zh hei}{SimHei}
\setCJKfamilyfont{zh kai}{KaiTi}
\setCJKfamilyfont{zh fs}{FangSong}
\setCJKfamilyfont{zh li}{LiSu}
\setCJKfamilyfont{zh you}{YouYuan}

\newcommand*{\songti}{\CJKfamily{zh song}} % 宋体
\newcommand*{\heiti}{\CJKfamily{zh hei}} % 黑体
\newcommand*{\kaishu}{\CJKfamily{zh kai}} % 楷书
\newcommand*{\fangsong}{\CJKfamily{zh fs}} % 仿宋
\newcommand*{\lishu}{\CJKfamily{zh li}} % 隶书
\newcommand*{\youyuan}{\CJKfamily{zh you}} % 幼圆
```

其它字体可以自由指定，SimSun，SimHei 这些指定字体的关键字可以直接用字体文件 *.ttf 来代替，也可以使用命令“fc-list”来具体查看系统中已安装的字体的名字，从而对应地填到上述命令中去。

2.1 Windows 系统

Windows 系统下可以直接安装 CTeX 发行的软件包，该发行包只有 32 位版本，64 位系统安装方法与 32 位略有不同。下面分别介绍。

2.1.1 32 位系统

包括 windows XP, Windows Server 2003, Vista, Windows Server 2008 和 Windows 7。

32 位系统下安装比较简单, 直接从 www.ctex.org 下载最新的安装包, 当前最新版为 2.9.2.164, 它包括 CTeX 环境, winEdt, MiKTeX, Ghostscript, 几个部分, 除 CTeX 环境外, 另外几个组件都可以选择安装, 如可以用 TeXLive 替代 MiKTeX, UltraEdit, gvim, Emacs 替代 winEdit 等。如果你不了解这几个软件, 那么就默认安装选项即可。

安装完后, 需要对 L^AT_EX 软件进行升级, 如 MiKTeX 软件进行升级, 升级的网络站点就选中科大的 CTAN 镜像站, 网址 <http://mirrors.ustc.edu.cn/CTAN>, 在教育网内速度还是比较快的, 如图 2.1, 2.2, 2.3 所示。

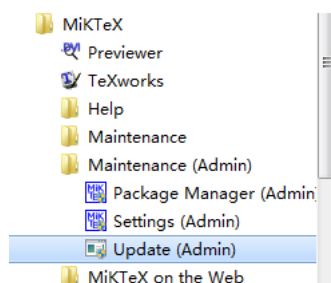


图 2.1 MiKTeX 升级设置

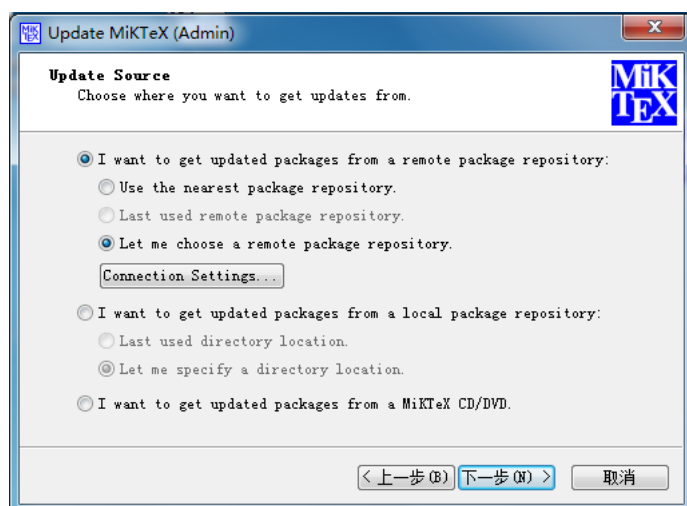


图 2.2 选择升级方式, 如图中选择一个升级站点

之所以要把软件升级到最新是因为该模版使用了最新的 hyperref 包中添加的命令 hidelinks。

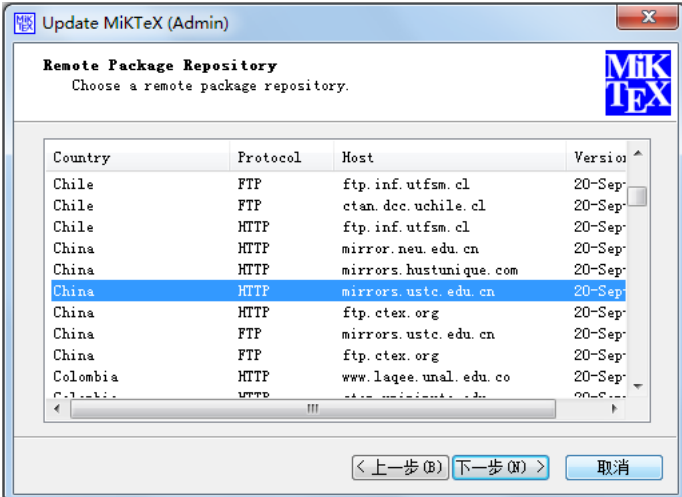


图 2.3 选择升级站点，以中科大镜像点为例

2.1.2 64 位系统

包括 windows XP 64bit, Windows Server 2003 64bit, Vista 64bit, Windows Server 2008 64bit, Windows 7 64bit 和 Windows Server 2008 R2。

安装 CTeX 安装包时，不要选择 MiKTeX，其它软件任意，与 32 位一样。安装完 CTeX 包后，去 www.miktex.org 网站下载 MiKTeX 的最新 64 位版，当前是 2.9 版，安装后在设置页面中把 CTeX 的目录加到 MiKTeX 的 Root 目录中去，如图 2.4。再如图 2.5 所示，刷新 MiKTeX 的目录数据库，就可以使用 CTeX 的环境了。



图 2.4 设置 MiKTeX 软件的 Root 目录

与 32 位 Windows 系统相同，安装完后也需要把 MiKTeX 升级到最新。

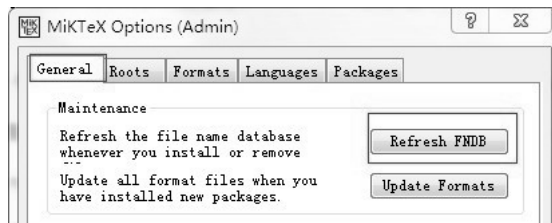


图 2.5 刷新 MiKTeX 的目录数据库

2.2 linux 系统

请自行选择 L^AT_EX 安装版本，然后把 C_TE_X 环境加入即可。我想用 linux 的应该都会这个吧。本次版本使用的操作系统及软件环境是 Debian 6 和 TeXLive2009。

请保证扩展包版本足够新。尤其是 hyperref 包，保证是 2011 年 8 月份后的版本。

2.3 Mac 系统

请自行选择 L^AT_EX 安装版本，加入 C_TE_X 环境。同 linux 下。

2.4 本模版所需要的扩展包

1. 图形、表格类扩展包 graphicx, array, booktabs-de, caption, natbib, multirow
2. 字体类扩展包 times(L^AT_EX 引擎中用), fontspec(X_YL_AT_EX引擎中用)
3. 目录选项扩展包 tocbibind, tocloft, makeidx, hyperref
4. 数学公式扩展包 amsmath, amsthm, amsfonts, amssymb, bm

2.5 测试运行

如果已经安装好了 C_TE_X 环境与 L^AT_EX 软件，那么，可以运行这个模版文件包里的 makethesis.bat 文件，几秒钟到十几秒后，如果生成了一份叫做“README.pdf”的文档，那么，恭喜，这个模版所需要的软件环境建立成功！如果没有生成这一份文件，那么有可能你的软件环境没有配置正确，比如把 L^AT_EX 软件升级到最新，这份模版所需要的扩展包没有被安装，请打开 L^AT_EX 软件自动升级功能，保证 L^AT_EX 软件能够成功地连接到 CTAN 站点下载所需的扩展功能包。

第3章 使用该模版

下面开始介绍如何使用这个模版，序言里已经提过，这个模版的目的就是让没有一点 \LaTeX 基础的同学也能很快地用 \LaTeX 组织出自己的论文。因此，这个模版没有过多的选项，只有一些和写 Word 同等难度¹的注意事项。但难点，要比 Word 少很多。

这个模板一共需要如下几个文件：

1. ZJUthesis.cls，这个是模版文件；
2. ZJUthesis.cfg，这个是模版的配置文件，与模版文件配合使用；
3. ZJUthesis.bst，这个是参考文献的格式说明文件；
4. 文件夹 CoverPagepic，存放着封面使用到的浙大校名与校徽图片；
5. CoverPagepic\ZJDX.eps，eps格式²的浙大校名图片；
6. CoverPagepic\ZJDX.pdf，pdf 格式³的浙大校名图片；
7. CoverPagepic\QSY.eps，eps 格式的校徽；
8. CoverPagepic\QSY.pdf，pdf 格式的校徽。
9. Chapter\Copyright.tex，这个是版权转让声明。
10. Signature\sign_ch.eps 等七个签名文件，这个是作者签名，默认是一个空白的 eps，提交最终版的时候可将其替换为相应的签名图片，直接可以作进 pdf 文档中去。这七个文件同样有其相对应的 pdf 格式图形文件，其文件名与相应签名对应如下：

sign_ch	作者的中文签名（中文题名页用）
sign_ch_s	导师的中文签名（中文题名页用）
sign_en	作者的英文签名（英文题名页用）
sign_en_s	导师的英文签名（英文题名页用）
sign_cr_1	作者的中文签名（版权声明页用）
sign_cr_2	作者的中文签名（版权声明页用）
sign_cr_s	导师的中文签名（版权声明页用）

¹是的，难度少很多很多。这是一个测试用的脚注

²关于 eps 格式，图片一节中将会做进一步介绍

³同 eps 解释

当然，其中几个签名可以用同一个文件复制使用。也可以做七个不同的使用。要注意的是签名的图形文件长宽比大约控制在 2:1。

现在即可以用 winEdt 或者其它习惯使用的文本编辑器，打开这个文档的 tex 源文件，就是叫做“论文模版示例.tex”的这个文件。我们书写论文，就是写这样一个扩展名为“tex”的纯文本文件。

我们现在开始它的第一行。

3.1 模版选项

```
\documentclass[oneside]{ZJUthesis}
```

这一句就指明了这个文档所用的格式模版，就是浙大的论文模版，**ZJUthesis** 就是这个模版文件的文件名。而 `\documentclass` 则是一个命令，在 \LaTeX 源文件中，以斜线 `\` 开头的字母字符串，都是一个命令，命令只能由一个斜线及其引领的字符串构成，不能包含数字与其它符号，当遇到非字母的其它字符时，该命令名字字符串结束。因此，`\documentclass` 是一个命令，不包含其它内容。该命令的参数，就是被包含在 `{}` 中的“ZJUthesis”，这个命令就是说明该文档使用的格式模版。而 `[oneside]` 则是 **ZJUthesis** 这个模版的参数。这个参数表示该论文现在采用单面印刷模式。

这个模版说明提供的可用选项只有两个：一个论文的单双面模式，另一个就是论文中链接的颜色。其中论文的单面模式是通过在 `\documentclass[oneside]{ZJUthesis}` 中插入的 `[oneside]` 来实现的，当把 `[oneside]` 换成 `[twoside]` 或者删除时，论文就变成了双面模式。

你可以按上面说的把这个说明文档改成双面模式，然后保存，再运行 `makethesis.bat`，生成出来的“论文模版示例.pdf”⁴，看看跟单面模式有何不同？

你想让论文是单面还是双面？选择就是这样简单！

在 tex 源文件中，以 `%` 起头的行都是注释行，在生成论文时，这些行将不会被论文包含，所以，可以利用注释行来写一些你对你的论文中一些文字的评注，比如一段内容暂时不放进去，不必删除它们，只需要把它们注释掉。就像这样：

```
% 这一段话我暂时先不放到论文里。  
这一段话将出现在论文里。
```

我们继续往下看：

⁴生成新的“论文模版示例.pdf”时，如果该文件正被打开，请先关闭。否则可能不会生成新的文件。”

这里我们看到了第二个参数，链接的颜色参数：

```
\hypersetup{colorlinks=false}
```

简单来说，就是这个模版会给这份文档中的链接，比如目录，索引，参考文献的编号加上颜色，以表示点击这个编号就能跳到相关的地方去。当然打印论文的时候我们并不想让它们有颜色，这个选项就是这个目的，如果你把“false”改成了“true”，那么保存后，再运行一下 makethesis.bat，看一下生成的“论文模版示例.pdf”中目录跟之前有何不同。

再往下就是文档的开始，用语句

```
\begin{document}
```

表示开始，

文档的最后有

```
\end{document}
```

表示整个文档结束相呼应

接下来

```
\fangsong
```

表示整个文档正文字体使用仿宋字体。小四字号已经在模版中设置好了，此处无须设置。

3.2 论文封面题名信息

封面已经在模版中制作过，只需填入相应信息即可。

<pre>\classification{TP311}</pre>	中图分类号，各专业分类号具体可以在 http://grs.zju.edu.cn/News/html/grs/xwsqjgf/xwsq/xwsq_bszn/2008-09-24/282-20080924085824.html 查询。
<pre>\serialnumber{10335}</pre>	单位代码，浙大是 10335。
<pre>\SecretLevel{绝密}</pre>	保密级别，如果没有，就不写这一句，封面上就不会出现保密级别。
<pre>\PersonalID{1234567}</pre>	申请号，一般是个人学号。
<pre>\title{大家好，我是论文名}</pre>	论文名。
<pre>\titletl{上在一行写不下}</pre>	如果论文名太长一行写不下，则分两行写，这里写第二行题目。如果一行就写得下，这一句不用出现。

如果论文名是两行，则请自行注意两行长度的分配。

中文论文题目： 我是第一行我比第二行长
我是第二行我短

中文论文题目： 我是第一行我短
我是第二行我比第一行长

这两种题目长度分配，请自行掌握。

<code>\englishtitle{Thesis Title}</code>	论文英文名
<code>\englishtitlel{Second Line}</code>	同样，如果论文名太长一行写不下，这里写第二行。如果一行写得下，这一句也不用出现。
<code>\Author{大名在此}</code>	姓名，请填上自己的名字。

如果想让名字各个字之间有个间距，如下所示效果：

申请人姓名： 王 东 举

则在填写姓名的时候按如下格式填写：

`\Author{王\hspace{1.5em}东\hspace{1.5em}举}`

其中的 `\hspace{1.5em}` 表示空间间距为一个半字符，如果要一个字符间距，则写 `1em`⁵，两个字符的间距，则是 `2em`，以此类推。

<code>\degree{某士}</code>	什么学位的论文，填“硕士”或者“博士”。
<code>\supervisor{导师姓名\space{1em}职称}</code>	填导师姓名与职称，填法与自己姓名填法类似，如果姓名中间要插入空白，请参考自己姓名中空白的插入法。
<code>\cpsupervisor{合作导师姓名\hspace{1em}职称}</code>	如果有合作导师，使用这个命令，没有，就不写这个命令，封面上内容会自动进行调整出不出现合作导师字样。
<code>\major{电气工程}</code>	请填写自己的专业名称，如字间增加空格，请参照姓名空格增加方法。
<code>\researchdm{研究方向}</code>	请填写自己的研究方向。
<code>\institute{电气工程学院}</code>	请填写自己的学院名称。
<code>\submitdate{2011年10月10日}</code>	这是论文提交日期，请自行填写。
<code>\defenddate{2011年11月1日}</code>	这是答辩日期，请自行填写，交草稿时此项不填，会自动留空。

到这里，封面内容填写完毕，可以使用生成封面的命令

`\CoverPagepic`

来生成封面。

3.3 实战：生成自己的论文第一页

看了上面的介绍，那么你是不是已经跃跃欲试准备将自己的论文从第一页开始了？下面就开始着手吧。

首先，先准备一个文件夹放与论文相关的所有文件，比如命名成“我的毕业论文”。然后，把“使用该模版”一节中提到的几个必须的文件都拷到“我的毕业论文”文件夹中去。接着，使用 WinEdt 或者其它文本编辑软件在“我的毕业论文”文件夹中建立一个扩

⁵`\hspace{1em}` 也可以写作 `\quad`。

展名为 `tex` 的文件，文件名自己取，我这里以“LATEX 论文模版使用说明”为文件名。最后，编辑“LATEX 论文模版使用说明.tex”为如下内容，具体元素内容请自行填写：

```
\documentclass{ZJUthesis}
\hypersetup{colorlinks=false}
\begin{document}
\classification{TM863}
\serialnumber{10335}
\SecretLevel{绝密}
\PersonalID{1234567}
\titletitel{论文题目}
\titletl{论文题目第二行}
\englishtitle{English Title}
\englishtitletl{Second Line}
\Author{名字}
\supervisor{导师名字}
\cpsupervisor{合作导师名字}
\major{专业名称}
\researchdm{研究方向}
\institute{所属学院}
\submitdate{提交日期}
\defenddate{答辩日期}
\makeCoverPage
\end{document}
```

以上内容可以直接从“论文模版示例.tex”中拷出来，建议保留其中的注释部分。编写 `tex` 源文件中，建议尽可能多做注释，以方便后期修改。

另一个注意的地方是，最后一行一定要是 `\end{document}`，你可以把“论文模版示例.tex”拉到最下面，看看它的最后是不是就是这样一句。

然后点击 WinEdt 工具栏中的 LaTeX 命令，大约几秒运行完成后，再点右边的 dvipdf 按钮，如图 3.1 所示，就可以生成“LATEX 论文模版使用说明.pdf”，打开就可以看到生成的封面了。

点“LaTeX”按钮的时候要注意的是，点右边的小箭头然后在下拉的菜单中只能是选中某个命令，此处选“LaTeX”，要执行要点到按钮上才行。

还可以创建一个批处理文件来完成这个任务，此处批处理文件内容如下：

```
latex --src-specials --synctex=-1 LATEX论文模版使用说明
dvipdfmx -p a4 LATEX论文模版使用说明
```

当前目录下运行这个批处理命令也可以行到“LATEX 论文模版使用说明.pdf”文件。

这里要说明的一点是，如果论文中开始加入参考文献，索引等信息，那么生成 pdf 文档的流程还要增加，使用 WinEdt 操作顺序是：先点击“LaTeX”命令按钮，再点击右边



图 3.1 运行 LaTeX

的“B”按钮（生成参考文献信息文件）和“I”按钮（生成索引信息文件）；然后**再**点击“LaTeX”命令**两次**，最后再进行 dvi 到 pdf 的转换，否则生成的 pdf 文件中参考文献与索引的引用处会是“？”号。如使用批处理命令生成，则相应的批处理文件内容应修改如下：

```
latex --src-specials --synctex=-1 LATEX论文模版使用说明
makeindex LATEX论文模版使用说明.idx
bibtex LATEX论文模版使用说明
latex --src-specials --synctex=-1 LATEX论文模版使用说明
latex --src-specials --synctex=-1 LATEX论文模版使用说明
dvi2pdfmx -p a4 LATEX论文模版使用说明
```

3.4 完成题名页信息

从上节已经完成了封面的信息输入与创建，下面我们进入题名页的创建，与创建封面页一样，题名页也是先输入信息，再创建页面。

还是回到“论文模版示例.tex”中，题名页分中文题名页与英文题名页。创建中文题名页要输入的信息有论文评阅人与答辩委员会名单，当然，论文在草稿阶段这几个信息是不填的，待到答辩完成，重新生成论文电子稿时，再填写这些信息。

论文评阅人信息代码及说明如下：

```
\reviewersA{姓\hspace{1em}名\hspace{1.5em}职称\hspace{1.5em}单位} 论文评阅人 1
\reviewersB{三字名\hspace{1.5em}职称\hspace{1.5em}单位} 论文评阅人 2
\reviewersC{三字名\hspace{1em}三字职\hspace{1em}单位} 论文评阅人 3
\reviewersD{姓\hspace{1em}名\hspace{1.5em}职称\hspace{1.5em}单位} 论文评阅人 4
\reviewersE{姓\hspace{1em}名\hspace{1.5em}职称\hspace{1.5em}单位} 论文评阅人 5
```

为了使这部分信息排版美观，这里要注意一点：

规划好“姓名”、“职称”、“单位”所占的空间，单位名称占用空间尽量不要超过十二个字。比如：

```
论文评阅人 1: 姓 名  职称  这个单位名比较长长长
论文评阅人 2: 三字名  职称  这个单位名短
论文评阅人 3: 三字名  副职称  这个单位名不很长
```

就要比

```
论文评阅人 1:  姓名  职称  这个单位名比较长长长
论文评阅人 2:  三字名  职称  这个单位名短
论文评阅人 3:  三字名  副职称  这个单位名不很长
```

看起来要好得多。
前者的代码如下：

```
\reviewersA{姓\hspace{1em}名\hspace{1.5em}职称\hspace{1.5em}这个单位名比较长长长}
\reviewersB{三字名\hspace{1.5em}职称\hspace{1.5em}这个单位名短\hspace{4em}}
\reviewersC{三字名\hspace{1em}副职称\hspace{1em}这个单位名不和很长\hspace{2em}}
```

注意 \spaceXem 的用法，一个 em 就是一个字宽。把所有单位名填充到同一长度，最长的单位名是 10 个字，6 个字的单位名就要补 4 个 em，8 个字的单位名补 2 个 em。请根据具体情况自行调整。

答辩委员会信息代码及说明如下：

```
\chairman{姓\hspace{1em}名\hspace{1.5em}职称\hspace{1.5em}单位} 答辩委员会主席
\commissionerA{三字名\hspace{1.5em}职称\hspace{1.5em}单位} 答辩委员会成员 1
\commissionerB{三字名\hspace{1em}三字职\hspace{1em}单位} 答辩委员会成员 2
\commissionerC{姓\hspace{1em}名\hspace{1.5em}职称\hspace{1.5em}单位} 答辩委员会成员 3
\commissionerD{姓\hspace{1em}名\hspace{1.5em}职称\hspace{1.5em}单位} 答辩委员会成员 4
\commissionerE{姓\hspace{1em}名\hspace{1.5em}职称\hspace{1.5em}单位} 答辩委员会成员 5
```

答辩委员会信息同样规划好姓名，职称，单位信息，规划方法与评阅人相同。

生成中文题名页

`\maketitle`

保存文件，生成 PDF 预览效果。

英文题名页信息输入与中文题名页类似。

此处英文题名需再输入一次，同样如果一行写不下写成两行。两行分配请自行处理。

<code>\Etitle{English Title}</code>	英文标题
<code>\Etitlel{Second Line}</code>	如果一行写不下，这里写第二行

英文题名页的评阅人与答辩委员会信息及说明如下：

<code>\EreviewersA{Name\hspace{1.5em}Professional Title\hspace{1.5em}Organization}</code>	论文评阅人 1
<code>\EreviewersB{Name\hspace{1.5em}Professional Title\hspace{1.5em}Organization}</code>	论文评阅人 2
<code>\EreviewersC{Name\hspace{1.5em}Professional Title\hspace{1.5em}Organization}</code>	论文评阅人 3
<code>\EreviewersD{Name\hspace{1.5em}Professional Title\hspace{1.5em}Organization}</code>	论文评阅人 4
<code>\EreviewersE{Name\hspace{1.5em}Professional Title\hspace{1.5em}Organization}</code>	论文评阅人 5
<code>\Echairman{Name\hspace{1.5em}Professional Title\hspace{1.5em}Organization}</code>	答辩委员会主席
<code>\EcommissionerA{Name\hspace{1.5em}Professional Title\hspace{1.5em}Organization}</code>	答辩委员会成员 1
<code>\EcommissionerB{Name\hspace{1.5em}Professional Title\hspace{1.5em}Organization}</code>	答辩委员会成员 2
<code>\EcommissionerC{Name\hspace{1.5em}Professional Title\hspace{1.5em}Organization}</code>	答辩委员会成员 3
<code>\EcommissionerD{Name\hspace{1.5em}Professional Title\hspace{1.5em}Organization}</code>	答辩委员会成员 4
<code>\EcommissionerE{Name\hspace{1.5em}Professional Title\hspace{1.5em}Organization}</code>	答辩委员会成员 5

同中文题名页类似，这里的人名，职称，单位信息尽量用简写，否则英文一行可能会写不下。另外，它们之间的间距请自行设计掌握，与中文信息间距设计方式一致。

至此，英文题名页信息输入完毕。用英文题名页命令生成英文题名页

`\makeEtitle`

这里不要忘记还有版权信息页

```
\SignautreDateA{2013}{10}{11}
\SignautreDateB{2013}{10}{11}
\SignautreDateC{2013}{10}{11}
\makeOSandCPRTpage
```

这里的三个时间命令对应版权信息页中的签字日期信息，在草稿及送审阶段，这几个命令可以不写。会自动把这几个位置留空。

3.5 正文部分各章节的书写

自此，论文的封面，题名页与版本信息页生成完毕，开始进入论文正文的书写阶段。下面的内容将不再以命令为主，而是以自己的论文内容为主。

还是回到“论文模版示例”，在正文部分的开始，我们看到这样一个命令：

```
\ZJUfrontmatter
```

这个命令的意思是开始论文的正文部分，将页码重置为大写罗马数字，并从 I 开始。

在谈各部分使用之前，写简单谈一下内容部分的书写。 \LaTeX 的文章内容书写没有什么特别的格式，直接书写即可。但需要注意以下几点：

1. \LaTeX 会忽略中文字符间的空格，也就是说：

大家好，我的中间没有空格。

和

大 家 好 ， 我 的 中 间 没 有 空 格 。

最后输出的内容是一样的。都是：

大家好，我的中间没有空格。

这个“中文字符”包括中文的标点符号，即“。”，“，”等。

如果需要在中文单词间插入空格怎么办？只需要用“~”来代替空格，就像这样：

大~家~好，~我~的~中~间~有~很~多~的~空~格~。

这样输出就是这样了：

大 家 好 ， 我 的 中 间 有 很 多 的 空 格 。

但在英文单词前后的空格， \LaTeX 会保留一个。

2. 断行与分段

\LaTeX 中，**单独一个回车**也会像空格一样被忽略，空行作为分段的标志，多个连续空行等效于一个空行。此外，还有一个断行命令，“\\”，或者是“\linebreak”，这个命令的作用是直接换行但不分段，即新换的行前不空两格。比如

我是第一行，
我还是第一行。

我是第一段，
我是第二段。

我是第一行，\\
我是第二行。

出来就是这个效果：

我是第一行，我还是第一行。

我是第一段，

我是第二段。

我是第一行，

我是第二行。

因此，在编写论文 `tex` 文件的时候，同一段落内可以任意换行，比如写一句话就换一行，这样有利于自己写的时候思路更清晰。只要不留空行，最后生成的文件都是一整段。

3. 中英文混合编辑时的一个问题及解决方案

如果某一段文字是中英文字符混合，比如含有单词，那么如果不做一点特殊处理，`LaTeX` 在生成 PDF 时可能会出现某一行最后一个英文单词突出行外。解决这个问题的办法就是在英文字符、单词的前面及后面加上空格符号。当然，这样不太符合我们自己的写作习惯，一个个加起来也太费事。没关系，`LaTeX` 的中文先驱们已经替我们简化了这个问题。CCT 套件中有一个程序：`cctspace.exe`，只要运行如下格式的命令：

```
cctspace [输入文件名] [输出文件名]
```

就可以执行这个空格插入操作，而无须手工一个一个添加。

这个程序在安装 `CTeX` 环境时已经被安装，可直接使用。使用 `linux` 的同学们，请自行在参考文献中的网站中下载源代码编译。

关于这个命令更多的功能选项说明，请参考张林波的《关于新版 CCT 的说明》^[3] 的 4.2 节部分内容，很短的，只有一页，纯中文说明文档。

4. 其它的一些要注意的内容

`LaTeX` 中有部分特殊符号不可直接输入，如下：

\$ % ^ & _ { } \

输入它们要用以下的命令

\# \\$ \% \^ \& _{} \{ \} \\$\backslash\$

以上就是书写论文正文内容时的一些说明，更多常见问题，请参考《一份不太简短的 \LaTeX 2 ϵ 介绍》^[1]和《CT \TeX FAQ》。这两个文档都可以在开始-菜单-程序-CT \TeX -help 中找到。

以下将分章节介绍各部分书写时如何使用该模版。

3.5.1 勘误页

其实很少有论文有这样一个部分，但作为一个部分，还是将其做到了模版内。如果需要，直接将其注释掉或者删除即可。

勘误页的调用命令为

```
\begin{corrigenda}
这里就写你的勘误内容。
\end{corrigenda}
```

有同学可能会发现，在“论文模版示例.tex”中这部分不是这么写的，我在接下来的致谢介绍中将解释原因。

3.5.2 致谢

致谢的调整命令与勘误页类似，为

```
\begin{thanks}
这里就写你的致谢内容。
\end{thanks}
```

但是看“论文模版示例.tex”中这一部分却只是简单写了一句：

```
\input{./Chapters/thanks}
```

这里的 `\input` 命令指输入另一个 `tex` 文件的内容到当前位置，就是说 \LaTeX 在编译这份 `tex` 源文件时读到这里，会去找这里指定的另一个 `tex` 文件，并把它的内容全部复制到这个位置。

这样做的好处是实现了论文模块化，可以把论文的不现章节写到不同的文件里去，每个文件都不会太长，更容易查看与修改。在这里这个命令所指向的文件就是当前目录中 `Chapters` 目录下的 `thanks.tex` 文件。同学们可以去看一下，这个 `thanks.tex` 文件中的内容，是不是就是跟我上面写的例子是一样的。

要注意的是，这里面的目录名与文件名不能为中文，否则会出错⁶。

3.5.3 序言

使用方法同致谢，代码如下：

```
\begin{preface}
这里就写你的序言的内容。
\end{preface}
```

3.5.4 摘要

摘要的填写与致谢类似，只多了一个关键字命令 `\keywords`，具体如下：

```
\begin{abstractC}
这里就写你的摘要的内容。

\keywords{关键字1, 关键字2}
\end{abstractC}
```

3.5.5 英文摘要

英文摘要的填法与中文摘要完全一样，只是都是用英文，具体如下：

```
\begin{abstractE}
这里就写你的摘要的内容。

\keywordsE{keywords1, keywords2}
\end{abstractC}
```

这个地方要注意的是与中文摘要命令不同之处，`abstractE` 与 `keywordsE`。

3.5.6 图片目录

这一部分只需要如下一个命令，其它的东西 \LaTeX 自己会替你搞定。

```
\ZJUListofFigures
```

如果你不需要图片目录，那么把这条命令删去或者注释掉。

⁶linux 下的同学可无视这条

3.5.7 表格目录

这一部分同样只需要如下一个命令，其它的东西 \LaTeX 会替你做好。

```
\ZJUListofTables
```

如果你不需要表格目录，同样只要把这条命令删去或者注释掉即可。

3.5.8 缩写、符号清单、术语表

这一部分与致谢部分相似。代码如下：

```
\begin{ListofSymbol}
这里就写你的缩写、符号清单、术语表的内容。
\end{ListofSymbol}
```

3.5.9 目录

这一部分只需要如下一个命令， \LaTeX 会替你搞定一切。

```
\ZJUcontents
```

3.5.10 正文章节内容

到这里，就到了正文章节的编写了。正文部分写法跟致谢，序言的相同，但致谢，序言没有更小的分节，也不会有图片，表格等内容。图片，表格方面内容在本说明中放在下面几章中，这里先只介绍正文的章节排布。

在开始正文内容之前，需要一个 \LaTeX 命令来告诉软件进入正文部分，页码需要重新从 1 编号，并使用阿拉伯数字，以及章节开始从 1 开始。这个命令是

```
\ZJUmainmatter
```

正文会有很多章，概述，条件，分析，讨论，结论什么的。编写 tex 文件的时候，需要给 \LaTeX 明确哪些是章，哪些是节，哪一些是更小的节。这样， \LaTeX 才能正确地给你的正文内容分章，分节

章节标题命令及使用如下：

```
\chapter{我是章的标题}
我是章标题下的内容
\section{我是第一节的标题}
我是节标题下的内容
\subsection{我是小节的标题}
我是小节的内容。
```

```
\begin{enumerate}
\item{我是并列项1}
\item{我是并列项2}
\end{enumerate}
\subsection{我是第二小节的标题}
我是第二小节的内容。
\section{我是第二节的标题}
我是第二节的内容
\subsection{我是第二节第一小节的标题}
我是第二节第一小节的内容
.....
```

章节标题中不需要说明是第几章第几节， \LaTeX 会替你计算，你所要做的只是把你的文章按结构统一起来即可。并且，目次也会根据你的标题内容自动生成。这里要补充一点的是：有的标题可能会比较长，写到目录里会导致目录条目换行，为解决这个问题， \LaTeX 允许目录中的标题与实际的标题不一样，要达到这个效果，以章的标题为例，其命令应加一个参数，如下：

```
\section[目录中的标题]{实际章节中的标题}
```

该模版中章节加上段落可以有 6 层，足够用了。如表 3.1 所示。

表 3.1 章节命令

命令	层次
<code>\chapter</code>	章 ^a
<code>\section</code>	节
<code>\subsection</code>	小节
<code>\subsubsection</code>	小小节
<code>\paragraph</code>	段
<code>\subparagraph</code>	分段

^a实际上在章的上层还有一层叫做 part（部分），本模版中用不到。

另外，如果某处需要用到几个并列项，可以采用 `enumerate` 命令，具体代码如下：

```
\begin{enumerate}

\item{我是并列项1}

我是并列项1的内容。

下面将有一个嵌套应用

\begin{enumerate}
\item{我是并列项1中的又一个并列项1}
```

我是小并列项1的内容。

```
\item{我是并列项2中的又一个并列项2}
```

我是小并列项2的内容。

```
\end{enumerate}
```

```
\item{我是并列项2}
```

我是并列项2的内容。

```
\end{enumerate}
```

生成文档的效果如下：

1. 我是并列项 1

我是并列项 1 的内容。

下面将有一个嵌套应用

(a) 我是并列项 1 中的又一个并列项 1

我是小并列项 1 的内容。

(b) 我是并列项 2 中的又一个并列项 2

我是小并列项 2 的内容。

2. 我是并列项 2

我是并列项 2 的内容。

注意 tex 代码部分中的空行。

这种并列项同样不需要考虑编号问题， \LaTeX 会替你做好一切。嵌套应用最多可以嵌套三层。

3.5.11 参考文献

正文自此完成了，下面的内容是后面的参考文献，索引，简历等部分。先需要一个命令告诉软件正文部分结束，开始文后部分

```
\ZJUbackmatter
```

当正文部分完成后，下一个部分就是参考文献。参考文献的生成需要准备另外一个参考文献数据文件 *.bib，本说明附带了一个数据文件例子：thesis.bib，该文件也是一个

纯文本文件，可以用记事本打开。里面给出了七种参考文献的格式：专利（`patent`）、标准（`standard`）、电子文档（`Edoc`）、期刊论文（`article`）、书籍（`book`）、博/硕士学位论文（`mastersthesis/phdthesis`）、其它文献（`misc`）。

下面以期刊论文为例来说明它的格式。

```
@article{
LUOZ:2007,
  Author = {罗振 and 田丰 and 孙小平 and 孙恩岩},
  Title = {基于{LATEX}的学位论文模板的设计与实现},
  Journal = {沈阳航空工业学院学报},
  Volume = {24},
  Number = {3},
  Pages = {45-48},
  Year = {2007},
  lang = {chinese}}
```

下面来逐行解释它们表示意义：

`@article` 表示这个文献信息所代表的文献类型是期刊文献。其它的文献类型见前两段内容。它后面紧跟的大括号 `{}` 将这个期刊文献的所有信息括起来。

第二行的“`LUOZ:2007`”是这个参考文献的引用名，可以起成任意的方便标识的字符组合，但不能使用中文。这个字符组合将在文中引用时使用。以这篇为例，当在论文中使用 `\cite{LUOZ:2007}` 时，在生成的文档中此处就会出现引用标号，具体标号 \LaTeX 会自己去计算，后面参考文献部分中的参考文献也由 \LaTeX 去排列。本模版设定的论文引用编号规则是按引用先后排序。

第三行到最后，每一行都代表了该文献的一项信息，分别是：作者（`Author`），题目（`Title`），期刊（`Journal`），卷号（`Volume`），期号（`Number`），页码（`Pages`），年份（`Year`），语言（`lang`），每项信息都被 `{}` 括了起来。要**注意**除了最后一项，每一项都以一个英文逗号“`,`”结束；多个作者之间用“`and`”连接；如果题目中有连续的大写字母，应用 `{}` 把它们单独括起来，如不括起来，除首单词首字母，其它都会被变回小写。

语言的信息项可以不写，这里写上是为了保障对中文文献信息的兼容。

其它几类文献的信息可参考 `thesis.bib` 中的填写方法，与上述说明大同小异。

当所有参考文献信息按 `thesis.bib` 中格式填写好，存为一个 `*.bib` 文件，文件名可以任意取，但不能使用中文，比如 `mybib.bib`。

在 `tex` 源文件中，只需在文中按上面介绍的 `\cite` 命令把引用参考文献的地方一一用标识词标示出来，再在正文的最后使用如下所示的参考文献列表命令，就可以生成格式完全符合要求的参考文献列表，这个格式符合要求包括任何一个标点都是正确的！！

```
\ZJUthesisbib{mybib}
```

请注意这里命令的参数就是你的参数文献数据文件的文件名。

3.5.12 附录

附录以命令`\appendix`开始，随后即可写入附录章节内容。附录章节内容写法与正文写法完全相同， \LaTeX 会替你处理好章节编号等问题，你完全不用考虑附录的编号。

```
\chapter{附录A}
\chapter{附录B}
```

3.5.13 索引

索引与参考文献的使用方法类似，但不需要建立什么数据文件，只要你在文中有索引标记，到了这里，只需要如下一条命令就能够建立索引。

```
\ZJUindex
```

3.5.14 个人简历

个人简历部分比较简单，按如下格式书写即可。相信写完论文后，这个完全难不住你们的。

```
\begin{resume}
\begin{enumerate}
\item{第一条的内容}
\item{第二条内容}
\end{enumerate}
\end{resume}
```

3.5.15 发表文章目录

发表文章目录部分同个人简历部分，命令只是关键字不同。

```
\begin{publications}
\begin{enumerate}
\item{第一篇}
\item{第二篇}
\end{enumerate}
\end{publications}
```

3.6 插入图片

论文中会插入图片、表格等元素。在 Word 中插图也是一件比较头疼的事，涉及图的位置布置，大小，清晰度等问题。在 \LaTeX 中这些问题也在一定程度上存在，但 \LaTeX 尽可能使得排版出来效果较好。相对于 Word，虽然不是所见所得形式，初期学习比较繁琐，但经短时间熟悉后可以比 Word 处理快不少。我最喜欢的方式就是用 Octave+gunplot 把曲线图做成标准一样的大小，然后把这些图命好名，放到一个专门的目录中去，一次全插好，如果要修改所有图的大小及式样，只要修改一下相应的 Octave 的生成曲线脚本文件即可。这样整篇文档中的曲线图都是统一的模式，统一大小，统一字体，非常整齐。

下面对 \LaTeX 中常见的插入图片方法作简要介绍，对于大部分专业做毕业论文应该是足够了，更多更复杂的方式，有兴趣的可以查找相关专题说明文档^[4]，这里提到的文献^[4]中给的是英文原版的下载地址，中文版搜一下网上到处有。

\LaTeX 支持的图片格式分两种情况：

1. tex->dvi->pdf 方式

包括 tex->dvi->ps->pdf 这样一个过程，本模版推荐采用的方式是 tex->dvi->pdf，这样的处理流程下，只能采用 eps 格式的图片。如果你只有 jpg, jpeg, png, gif 之类格式的图片，那么不要紧， \LaTeX 中提供了一个转换程序 bmeps.exe，可以帮助你吧 jpeg 图片转换成 eps 格式，当然你也可以用 photoshop, gimp 转换。使用 bmeps 的命令格式如下：

```
bmeps -p 1 -c picture.jpeg picture.eps
```

其中，-p 1 是压缩出的图片质量，1 最好，2 是默认，3 最差。-c 是保留彩色，不带 c 图片为黑白。

2. tex->pdf 方式

如果是用 pdfTeX 直接将 tex 转换为 pdf 的流程，则不能采用 eps 格式，但可以采用 jpg, jpeg, png, gif 这些格式。

如果只有 eps 格式图片，则可以使用 photoshop, gimp 或者 Acrobat 将其转为 pdf 格式。

这也是为什么这个模版带的文件中既有 eps 又有 pdf 的原因，为了照顾到部分直接将 tex 转换到 pdf 的同学。

本模版推荐新手使用 eps 格式图片。eps 格式没使用过 L^AT_EX 可能会比较陌生, eps 也是一种矢量格式图形。就是说如果是从矢量图形转到 eps 格式, 一样可以无损缩放。

subsection 图片插入的一般方法

在文中插入图片的常用命令如下 (以使用 eps 格式为例):

```
\begin{figure}[htb]
\centering
\includegraphics{picture.eps}
\caption{图片标题}
\label{Figflag}
\end{figure}
```

`\begin{figure}[htb]` 是告诉 L^AT_EX 软件准备插入图片, `[htb]` 是一个选项, 代表图片的插入位置, `h` 代表在文中当前位置, `t` 代表一页的顶部, `b` 代表页面底部, 还有一个选项是 `p`, 代表图片要单独放在一页中, 如果不给出这些选项, L^AT_EX 会试着依次选用 `h-t-b-p` 方式, 找出自己认为最合适的, 如果给出一选项, L^AT_EX 就只使用给出的一个选项, 如果给出几个选项, L^AT_EX 会尝试给出的这几个选项, 然后选用它认为最合适的。一般采用默认方式, 不给出任何选项。除非有个把图 L^AT_EX 自动给出的位置不好, 再通过这几个选项来选择。

`\centering` 表示图片的位置为居中放置。

`\includegraphics{picture.eps}` 指出了要添加的图片的文件名, 如果图片在其它目录中, 比如 `pictures` 文件夹, 命令则变为:

```
\includegraphics{pictures/picture.eps}。
```

要着重说明的是, 插入的图片可以采用选项进行缩放, 旋转。比如, 要把图片设置为同文字部分一样宽, 如图 3.2, 则命令变为:

```
\includegraphics[width=\textwidth]{picture.eps}
```

指定为与文字一样宽太宽了, 下面来把它缩小点, 比如半个文字宽度, 那么就用 `width=0.5\textwidth`, 如果指定 10cm 宽, 就用 `width=10cm`, 如果指定 7cm 高, 就用 `height=7cm`, 如果是指定原图大小的 50%, 就用 `scale=0.5`, 如果我们想把它调整到原尺寸大小的 50%, 再逆时针旋转 90 度 (如想顺时针旋转则使用负角度值), 就用如下的命令:

```
\includegraphics[scale=0.5,angle=90]{picture.eps}
```

效果如图 3.3 所示。

常用的图片调整参数有: `width`, `height`, `angle`, `scale` 这四个, 如果想要更多的效果, 比如把图片四边裁掉, 像 Word 中那样。请参考 `graphicx` 包的帮助文件 `graphicx.pdf`⁷。

⁷此处要说明一下, L^AT_EX 安装的所有扩展包都有对应的说明文档, 请到 L^AT_EX 的安装目录中寻找, 比如本模版使用的



图 3.2 跟文字一样宽的图

`\caption{图片标题}` 给出了图名，使用这条命令后，图下方就有了图名。并且图片目录中也会出现相应的内容。如果图的名字比较长，有可能会造成图片目录中的目录项换行，类似章节标题命令的格式，增加一个参数哪下：

`\caption[图片目录中的标题]{图片下方出现的标题}`

`\label{FigHalf90}` 这里给出了这个图片的标签“FigHalf90”，当你文档中要出现“如图 X.x 所示”时，只要写成“如图 `\ref{相应的图片标签}` 所示”即可。L^AT_EX 会替你

MiKTeX2.9 的安装文件夹。帮助文件文件名一般与扩展包名相同，格式为 pdf 或者 dvi，少量的扩展包帮助文件直接给的 tex 源文件，需要自行生成文档。



图 3.3 原图一半大小并逆时针旋转 90 度的图

把标签换成图的编号。这里要注意的是，一是引用命令前后各有一个空格，二是标签不能含有中文字符。

`\end{figure}` 表示图片插入结束。

这里提一些题外话，无兴趣研究 \LaTeX 的同学请无视此段，以上使用的图片插入命令并不符合 \LaTeX 命令标准，但是因为其方便应用，成为了应用最广泛的 \LaTeX 命令。标准的命令格式在 `graphicx.pdf` 中有详细说明。另外， \LaTeX 还有一个图形扩展包，叫做 `graphics`，注意最后一个字母，一字之差，`graphics` 包支持的功能更多一些，但大部分普通应用应用不到，而且，`graphics` 只支持标准命令调用。有兴趣的同学可以研究一下 `graphics.pdf`，此处我就不再多说了。

3.6.1 插入图片的应用进阶

刚才前面有提到：`eps` 是一种矢量格式图形，所谓矢量格式图形，就是放大缩小都会造成图片变“糊”。其实，`eps` 的方便之处还不止这些，大部分的科学计算软件都支持生成 `eps` 格式图片，比如 `matlab`，可以直接生成 `eps` 格式图片。

有同学会说，那我用 `Microsoft Visio` 画的图怎么处理呢？`Visio` 创建的图形也是矢量图，在 `Word` 下是可以任意缩放的。如果你装有 `ps` 虚拟打印机或者 `acrobat`，这个问题就迎刃而解了。首先选中你要打印的 `visio` 图形，然后把它打印成 `ps` 或者 `pdf` 文档。接下来 `ps` 文档的话就直接用安装 `CTeX` 组件时带的 `Ghostscript` 转换成 `EPS` 图像，`pdf` 文档的话就先用 `Acrobat` 打开，菜单 - 工具 - 高级编辑 - 裁剪工具，把所需要的图形那一部分剪出来，另存为 `eps` 文件即可。用 `photoshop` 同样可以达到这个效果。这样出来的图像依然是矢量图，可以任意缩放不会糊。

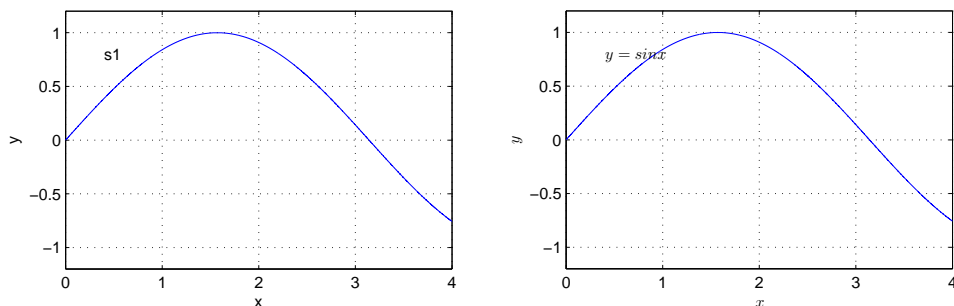


图 3.4 matlab 作图例

\LaTeX 通过使用 `psfrag` 扩展功能包，还能对 `eps` 中的一些字符进行替换。但是遗憾的是，`psfrag` 只支持 `dvips` 转换程序，本模版推荐使用的 `dvipdfm` 程序不支持该扩展包。但这不要紧，我们可以采用变通的方法，把要替换字符的图用 `tex->dvi->ps->pdf` 的流程处理，再生成的 `pdf` 用 `Acrobat` 裁剪后生成 `eps` 文件即可。

参考代码如下，生成图片效果对比如图 3.4 所示：

```
\documentclass{article}
\usepackage{graphicx}
\usepackage{psfrag}
\begin{document}
\pagestyle{empty}
\psfrag{s1}{$y=\sin x$}
\psfrag{x}{$x$}
\psfrag{y}{$y$}
\includegraphics{matlabfig.eps}
\end{document}
```

图 3.4 中左图为 `matlab` 生成的 `eps` 图，右图为将左图中的 `s1`，纵横坐标名替换后的图片。可以把这一页放大几倍来看，图片依然非常清晰！这就是矢量图的优势。替换文字中如果有中文出现，因为 `dvips` 程序与 `cjk` 不兼容，本论文模版的 `tex-dvi-pdf` 生成方式将不能用于含有 `dvips` 流程，使用 `dvips` 须使用 `cct` 方式，具体请参考 `CTeX FAQ`，有详细例子。

3.7 插入表格

\LaTeX 中建立表格比 `Word` 稍复杂一些，但表格的格式调整及效果比 `Word` 简单得多。

3.7.1 普通表格的插入

一个普通的插入表格代码如下所示，其生成的表格如表 3.2 所示：

```

{
\begin{table}[htb]
\zihao{5}
\caption{表格标题}
\label{Tabkeyword}
\centering
\begin{tabular}[t]{c|l|r|p{4cm}}
\hline
居中 & 靠左 & 靠右 & 靠左宽4cm\\
\hline
Center & Left & Right & Width=4cm\\
\hline
\end{tabular}
\end{table}
}

```

表 3.2 表格标题

居中	靠左	靠右	靠左宽 4cm
Center	Left	Right	Width=4cm

表格的使用一部分与图片相类似，比如开始的语句`\begin{table}[htb]`，以及结束的语句`\end{table}`，与图片语句相比，除了把“figure”换成了“table”，其它都相同。语句的含义也相同。

`\zihao{5}` 是把表格的字号改小一号，一般而言，表格字号应比正文字号小一号，论文中的正文字号是小四号，那么表格中的字号就应该是五号字体。这个命令一定要写在`\begin{table}`的下方才能对该表格起作用。

`\caption{表格标题}` 是表格的标题，与图片标题格式相同，会自动加到表格目录中去，如果表格标题太长，可以用`\caption[目录中的标题]{正文中的表格标题}`这种形式。

`\label{Tablekeyword}` 与图片的标签相同，只要在正文中使用`\ref{Tablekeyword}` 就可以在生成的文档中显示出表的编号，同样，命令前后要各留一个空格，并且标签不能使用中文字符。

`\centering` 使整个表格居中放置。

`\begin{tabular}[t]{c|l|r|p{4cm}}` 是表格开始的标志，注意与`\begin{table}`区别开来，`[t]` 表示表格与其上方的正文行对齐方式是行顶对齐，如果是`[b]` 则为行底对齐，这两种对齐方式可以在实际操作中试一下，再做选择。`{c|l|r|p{4cm}}` 则代表这个表格各列的对齐方式，`c` 为居中，`l` 为左对齐，`r` 为右对齐，`p{4cm}` 为该列宽度为 4cm，并且左对齐。中间竖线“|”则表示表格使用竖分隔线，如果没有这个竖线符号，对应位置就没有分隔线，比如表 3.2 就没有最左边和最右边的竖线。如果写为`{|c|l|r|p{10cm}|}`，

就有最左边和最右边的竖线了。如果不需要竖分隔线，则写为`{c|lrp{10cm}}`即可。要使用双竖线分隔线，就写为`{c||c}`；还可以使用其它字条代替，比如本模版的题名页中评阅人及答辩委员会信息就是利用的表格形式，它的分隔符是冒号“:”，相应的命令就是`{c@{: }c}`，即`@{: }`。

`\hline` 就是画一条横线，如果没有写这一句命令，你会看到表格没有横分隔线。

表格内容是一行一行来填的，每一行结束使用断行符`\\`，同一行中不同列的元素用 `&` 来分开。

`\end{tabular}` 表示表格结束。

3.7.2 把表格换个样式

上一小节介绍的方法生成的表格一般情况下可以用了，但 \LaTeX 的能力不止是做这样一个常规表格，下面我们来利用 `booktabs-de` 扩展包来让表格看起来更 professional 一些。这个扩展包的调用已经放在了本模版中了，使用本模版可以直接使用这种格式。这个表格如表 3.3 所示。

表 3.3 粗细线表格示例

长 (m)	宽 (cm)	高 (mm)	重量 (kg)
1.234	5.676	332.876	3498.5
548.4	23.43	34.98	923.8

表 3.3 的实现代码如下所示：

```
{
\linespread{1}
\begin{table}[htb]
\zihao{5}
\caption{粗细线表格示例}
\label{Tab3Line}
\centering
\begin{tabular}[b]{cccc}
\toprule
长 & 宽 & 高 & 重量\\
(m) & (cm) & (mm) & (kg)\\
\toprule
1.234 & 5.676 & 332.876 & 3498.5\\
\midrule
548.4 & 23.43 & 34.98 & 923.8\\
\bottomrule
\end{tabular}
\end{table}
}
```

先解释一下这部分代码为什么会用一对大括号 `{}` 整个括起来。如果要对文章中一部分内容的格式作调整，又不能影响到其它部分，就使用一对大括号把要调整格式的部分括起来。这里要调整的格式是行距。本论文的默认行距是 1.5 倍行距⁸，在这里要把它调成单倍行距这个表格样式才比较美观，因此就在最前使用 `\linespread{1}` 命令把它的行距调整为单倍行距。关于局部格式调整，在后面的《重点字符，醒目提醒》一节中还将详细介绍，这里只对牵涉部分做简要说明。

这部分代码与普通表格代码的区别在于横线生成命令：`\toprule`，`\bottomrule` 都是比较粗的线条，`\midrule` 是比较细的线条。此外，还有一个不完整线条命令，`\cmidrule`，该命令的使用类似下一小节要讲到的 `\cline` 命令。

3.7.3 有单元格合并的表格

表格不可能总是一格归一格的。部分表格总是需要进行表格合并，下面我就举一个表格合并的例子。`LaTeX` 直接支持列合并，行合并则需要扩展包 `multirow` 的支持，本模版已经将该扩展包包含进去，可直接使用。如表 3.4 所示表格

表 3.4 复杂表格示例

我占了两列		第 3 列	第 4 列	第 5 列
我占了两行	第二行第 2 列	第二行第 3 列	我占了两行又两列	
	第三行第 2 列	第三行第 3 列		

表 3.4 的实现代码如下：

```
\begin{table}[htp]
\zihao{5}
\caption{复杂表格示例}
\label{TabComplex}
\centering
\begin{tabular}[t]{|c|c|c|c|c|}
\hline
\multicolumn{2}{|c|}{我占了两列} & 第3列 & 第4列 & 第5列 \\
\hline
\multirow{2}{*}{我占了两行} & 第二行第2列 & 第二行第3列 & \multicolumn{2}{*}{我占了两行又两列} \\
\cline{2-3}
& 第三行第2列 & 第三行第3列 & \multicolumn{2}{*} \\
\hline
\end{tabular}
\end{table}
```

⁸模版中的行距调整命令是 `\linespread{1.5}`，但并不是直接的 1.5，这里有个换算，有兴趣的同学可以查看参考文献^[1]

`\multicolumn{2}{|c|}{表格内容}` 就是合并列的命令，参数 2 表示合并两列，格式命令表示合并后的单元格居中对齐且两边有分隔竖线，注意在有分隔线时，分隔线符号 | 不要忘记。

`\multirow{2}*{表格内容}` 是合并行的命令，参数 2 表示合并两行，注意中间的 * 号。

注意同时合并行合并列时两个命令的嵌套方式，以及合并行情况下，除第一行外其它行的表示方式。如合并两行两列时第二行的 `\multicolumn{2}{|c|}{}`。

`\cline{2-3}` 代表横线只划第二列与第三列，如果划 2,3,5 列，则写为 `\cline{2-3,5}` `\cmidrule` 用法同 `\cline`。

3.7.4 其它高级表格格式及扩展包简介

如果需要通过实现表格左上角内有斜线的样式，请使用 `slashbox` 扩展包并参考其自带帮助文档。

更复杂的表格，比如指定每一行的宽度，即可以借助盒子^[1]来完成，也可以借助扩展包 `array`，`tabularx` 等，具体实现代码此处不再详述，有兴趣的同学可参考文献^[1,5]以及扩展包内帮助文件。

3.8 公式及其编号

\LaTeX 出现之初的重要作用就是排版各种复杂的数学公式，关于数学公式的排版方法，参考文献^[1]中有一整章的内容来介绍各种公式的编辑。我这里就偷个懒不写了，太长了，而且公式编辑我写不出我的特色来。

\LaTeX 的公式编辑效果远胜于 Word 中的公式编辑器，而且，使用 \LaTeX ，完全不用担心公式编辑器的版本问题，因为，这里公式就是用纯文本写的。下面我举一个小小的公式例子来说明公式的编写及编号，如式 3.1 所示。

$$c^2 = a^2 + b^2 \quad \text{公式 (3.1)}$$

实现式 3.1 的代码如下所示：

```
\begin{equation}
\label{Equ:exm}
c^2=a^2+b^2
\end{equation}
```

`\begin{equation}` 与图片，表格很类似，开头都是一个环境开始命令，只是这里的关键词是 `equation`。

`label{Equ:exm}` 是给公式一个编号，以便在文中引用。该公式编号由 \LaTeX 负责，一般不需人工干预。

`\ref{Equ:exm}` 就是文中引用公式的命令。

3.9 索引

索引比较简单，只要在你想放索引的位置放上命令 `\index{索引名}`，然后文后面的索引列表中就出现“索引名”这个条目，并且有它的页码，索引支持中文，英文字符。如果文中有多处同名索引，这个表中都会把它们按字母表顺序列出来。很方便的，你想在哪里插入索引就在哪里插入索引，完全不必考虑索引列表的事。

3.10 参考文献

前面介绍参考文献时就提到，只要准备好自己的参考文献数据库，在引用参考文献时只需要利用 `\cite{tag}` 就能把相应的参考文献引上。这里的 `tag` 就是在参考文献数据库中各参考文献定义的标签，不能含有中文字符。多个文献，则中间用英文逗号分开如 `\cite{tag1,tag2}`。不需要考虑引用顺序， \LaTeX 会替你打理好一切。

如果想用形如“文献 [1] 提出 XXX”的格式，则使用如下代码。

文献 [`\citenum{tag}`] 提出

该模版已经把各类参考文献的列出格式按毕业论文要求调整好，不需要再修改。

如果你想了解更多参考文献格式调整的信息，请参考扩展包 `natbib`，`custom-bib` 自带的帮助文档。本次修订版新增的第 5 章也专门会对参考文献的格式校调问题进行大篇幅介绍。

3.11 脚注

有些时候需要使用脚注， \LaTeX 的脚注功能也是很方便的，只要在你想要放一个脚注的地方用命令 `\footnote{脚注内容}` 即可，脚注内容长度不限，脚注编号 \LaTeX 会替你操心。你要做的，只是写好这个脚注的内容。

对于表格加脚注，就像表 3.1 显示效果那样，则稍显麻烦一点，需要把表格放进一个 `minipage` 里去。关键代码如下：

```
\begin{table}
.....
\centering
\begin{minipage}[c]{10cm}
\centering
\begin{tabular}
.....
\end{tabular}
\end{minipage}
.....
\end{table}
```

`\begin{minipage}[c]{10cm}` 表示创建一个内部居中对齐的，宽度为 10cm 的小页面，这个宽度要根据其内部表格宽度来确定，这样出来的脚注才美观。

3.12 重点字符，醒目标示

有的时候需要对某些内容做一个醒目标示，比如改变字体啦，字体加粗啦。这个时候首先要把需要改变格式的部分用一对大括号⁹括起来。然后在头上写上字体字号之类的命令。就像这样

我这一段话里有**加粗**，有**楷体并加粗**，有**黑体**，还有**三号字**字体。

这一句话的实现代码如下：

我这一段话里有`\bfseries` 加粗}，`\{kaishu\bfseries` 有楷体并加粗}，有`\{heiti` 黑体}，还有`\{zihao{3}` 三号字}字体。

在 \LaTeX 中可以用的字体有宋体`\songti`，仿宋`\fangsong`，楷体`\kaishu`，黑体`\heiti`，幼圆`\youyuan`，隶书`\lishu` 六种。字号命令是`\zihao{1}` 字号从 1，-1，2，-2，一直到 8 号字，前面带负号表示小号字，比如 -4 就是小四号字体。7，8 两个字号没有小号字对应。

关于加粗字体命令`\bfseries`，对英文字符只要使用`\bf` 即可，中文字符应使用`\bfseries`。

关于字体效果的设置，更多内容参见 CTeX 自带的帮助文档 `ctex.pdf`。

如果想使用更多的字体，比如微软雅黑，请自行网络搜索方法，自己生成字体文件并关联上即可。CTeX 提供的六种字体，一般情况下已经足够使用。

⁹要使用英文字符的大括号 `{ }`

3.13 插入算法伪代码

以下是插入算法伪代码示例，如算法1所示。

Algorithm 1 基于距离变换的骨架提取

Require: 前景的二值图 bw

▷ 像素的灰度值为 0 或 1

Ensure: 骨架图 $skel$

```

1: // 第 1 次遍历：从上往下，从左往右
2: for  $i = 1, \dots, M$  do                                ▷  $M$  是二值图的高度
3:     for  $j = 1, \dots, N$  do                                ▷  $N$  是二值图的宽度
4:          $bw[i][j] = 1 + \min(bw[i][j-1], bw[i-1][j])$         ▷  $\min$  函数取极小值
5:     end for
6: end for
7: // 第 2 次遍历：从下往上，从右往左
8: for  $i = M, \dots, 1$  do
9:     for  $j = N, \dots, 1$  do
10:         $bw[i][j] = 1 + \min(bw[i][j], bw[i+1][j], bw[i][j+1])$ 
11:    end for
12: end for
13: // 第 3 次遍历：获取骨架图
14:  $skel$  的空间分配，并将每个像素初始化为 0
15: for  $i = 1, \dots, M$  do
16:     for  $j = 1, \dots, N$  do
17:         $t = \max(bw[i-1][j], bw[i+1][j], bw[i][j-1], bw[i][j+1])$     ▷  $\max$  函数取极大值
18:         $t = \max(t, bw[i-1][j-1], bw[i-1][j+1], bw[i+1][j-1], bw[i+1][j+1])$ 
19:        if  $bw[i][j] \geq t$  then
20:             $skel[i][j] = 1$                                 ▷ 骨架点
21:        else
22:             $skel[i][j] = 0$ 
23:        end if
24:    end for
25: end for
  
```

3.14 插入源代码

代码示例:

```
1 int main(int argc, char ** argv)
2 {
3     printf("Hello world! \n");
4     return 0;
5 }
```

3.15 插入定义、定理等

实例来自 <https://github.com/eclipselu/zjuthesis-mphil>。

假设 3.1: 待月西厢下，迎风户半开；隔墙花影动，疑是玉人来。

$$c = a^2 - b^2$$

公式 (3.2)

$$= (a + b)(a - b)$$

公式 (3.3)

定义 3.1: 子曰：「道千乘之国，敬事而信，节用而爱人，使民以时。」

定理 3.1: 犯我强汉者，虽远必诛。

——陈汤（汉）

证明 3.1: 天不言自高，水不言自流。

$$\begin{aligned}\varphi(x,z) &= z - \gamma_{10}x - \gamma_{mn}x^mz^n \\ &= z - Mr^{-1}x - Mr^{-(m+n)}x^mz^n\end{aligned}$$

$$\zeta^0 = (\xi^0)^2,$$

公式 (3.4)

$$\zeta^1 = \xi^0\xi^1,$$

公式 (3.5)

$$\zeta^2 = (\xi^1)^2,$$

公式 (3.6)

第4章 参考文献格式处理

参考文献是学位论文中重要的一部分，规范、整齐的参考文献引用格式，对毕业论文质量有重要的影响。

L^AT_EX 提供了一套比较全面的参考文献辅助系统，涉及参考文献引用格式、文后列写格式两大方面，其中引用格式又细分为按序号、按人名年代，按人名字母顺序等几小类索引方式，文后列写格式除以上所述索引方式外，还牵涉外文人名书写格式、文献信息排列方式、文献信息条目格式等几方面。如果应用得当，则 L^AT_EX 将成为整理参考文献的顶级帮手，做出非常漂亮统一的参考文献引用内容。但是，L^AT_EX 的参考文献引用方法又是网上各种资料薄弱的部分，除了文献 [1] 中提到的方式外，其它找得到的中文资料寥寥无几，系统介绍其使用方式方法的文献更是几乎没有。这个模版的第一版中，对参考文献部分只做了非常简单的说明，只能照着搬来使用，但并不能自主进行自由修改。这一章以下部分将对 L^AT_EX 参考文献系统的使用做一个详细的说明。

4.1 一般处理方案

上面已经说过，关于 L^AT_EX 的参考文献系统，主要是参考文献 [1] 中提到了一种最简易的方法。即利用 thebibliography 环境将参考文献条目一条一条地列出来，有点类似于在一般在 Word 中的处理方式¹，也是在最后面一条一条地把参考文献抄写上去。而且参考文献信息中各个元素的格式也是在抄写中一条一条写进去。

这种方式的好处是简单易行，适用于比较小的文档，参考文献不超过二十个的文章，但对于毕业论文这种大工程而言，这种方式就有些力不从心了。就需要有一种适合于处理大批量，多类型参考文献的方式与工具。

4.2 专业处理方案

L^AT_EX 系统中提供了 BibTeX 这样一人专门处理参考文献的工具，这个工具将人工一条一条的罗列变成了机器自动整理格式，自动罗列，但是，因为参考文献格式多种多样，这个工具的也需要有很广泛的适用性，于是，它被做得很复杂，以至于除了提供的默认方式外，很少有人知道如何去改动它，调整它的行为方式。

¹用 NoteExpress, EndNote 软件的另说，L^AT_EX 处理参考文献其实也是跟这类软件类似的思路。

4.3 参考资料

前面已经提到， \LaTeX 提供了一套比较全面的参考文献辅助系统，按应用范围分为文中引用格式和文后罗列格式两大部分，这两大部分**互相独立**，其联接的纽带就是被称为“key”的自定义字符串。

关于这两部分功能使用的资料，介绍文中引用格式的比较多，介绍文后罗列格式的比较少。

\LaTeX 发行版中自带的参考文档已经把这两个问题做了系统的阐述，只是文档是英文的，没有中文资料，而且文后罗列功能介绍部分写得又比较晦涩难懂。

文中引用格式部分，参考资料是 `natbib.pdf`^[6]。就是 `natbib` 功能扩展包的包说明文档。这份资料的大意是根据文后罗列的条目内容，在文中引用是用数字序号，还是用作者年代的方式。作者年代这种序号方式其实适用于拉丁字母系文献，对中文文献而言还是数字序号比较适用。这个模版的中就使用了这个 `natbib` 功能包的功能，具体设置可以打开 `cls` 文件查看我设置的几个选项，相关选项含义可以参考该文献。这部分内容比较好理解，难度相对较低。

文后罗列部分的资料比较复杂，最普通的前面讲的文献 [1] 中就有。而关于 `BibTeX` 的使用方式的，则需要以下四个在 \LaTeX 发行包中的帮助文档：`merlin.pdf`^[7]，`makebst.pdf`^[8]，`btchhak.pdf`^[9]，`btchdoc.pdf`^[10]。其作用介绍如下：

- `merlin.pdf` 是介绍 `BibTeX` 的配置文件可配置选项的；
- `makebst.pdf` 是介绍 `BibTeX` 的配置文件结构构成的；
- `btchhak.pdf` 是介绍 `BibTeX` 配置文件中的逆波兰代码基本语法的；
- `btchdoc.pdf` 是介绍 `BibTeX` 默认支持的参考文献类型及其必要信息结构的。

4.4 \LaTeX 参考文献组织思路

前面也已经提到过， \LaTeX 把参考文献任务分成了两部分，一个是文中的引用符号填写任务，另一个则是文后附带的参考文献信息的罗列部分。这两部分只有在序号或者说是排序的时候由用户指定的“key”来进行关联指定。这样一来，程序的任务就比较明确了。

首先，先扫描文中参考文献引用，在相应的参考文献库中找到相对应的“key”的条目，记录下来。然后，根据扫描文中参考文献的引用结果，将相应的参考文献条目挑出来，

根据其设定的顺序（引用顺序，或者是作者年代顺序），将相对应的参考文件信处条目，BibTeX 按照设定的格式列写出来，再由 L^AT_EX 将其写到文后中去。如果打开这个模版对应的“论文模版示例.bbl”文件，可以看到，实际上是 BibTeX 帮你写了一个 thebibliography 环境，L^AT_EX 还是按照一般的 thebibliography 环境方式对其进行调用，写入最终生成的 pdf 文档中去。也就是说 BibTeX 这里充当了一个有规则的抄写员的作用。

4.5 如何让 BibTeX 按你的想法干活

如果文后面的参考文献列表不是自己采用 thebibliography 环境生成，那就是是由 BibTeX 来生成的，BibTeX 生成这样一个参考文献列表的基础信息来自提供的 *.bib 文件，这里面用通用的格式写出了各条参考文献的信息，如作者，题名，年代，出版者，出版地等，也标明了各条参考文献的类型，如期刊文章（article），书籍（book），电子出版物（EPublication）等。有了这些信息，还需要告诉 BibTeX 要求输出的格式，BibTeX 才能按照要求写出条目。那么问题就是：如何告诉 BibTeX 这些格式信息呢？

答案就是 *.bst 文件，这个文件就是 BibTeX 的配置文件，它告诉了 BibTeX 对各种文献类型，文献信息的处理方式。但这个文件是整个参考文献系统中最复杂也是最核心的部分，我在这上面花了很长的时间摸索才找到关键所在。在写第一版的时候，仅仅是根据从网上找到的很少的英文论坛中的几句话，硬改出了一个接近于浙江大学学位论文要求的参考文献格式，在这次的改版中，则完全根据其设定规则，按照其设置方式，进行了修改设定。

4.6 关键所在：*.bst 文件

正因为 bst 文件的规则与内容是整个参考文献系统中最复杂的部分，因此针对这个问题，降低使用者的难度，BibTeX 的开发者们就提代了一套默认的选项来给我们用。如自带的 plain.bst，natbib 包提供的 plainnat.bst，abbrvnat.bst，unsrtnat.bst，一般情况下，不是特别的有要求的，这些默认的 bst 文件也是勉强够用的。

之所以说是勉强够用，是因为提供这些模版的，都是外国人，他们的行事方式，文献组织方式终归与中国人有一定的不同。如文献信息罗列的次序，文献类型的说明，都与中国要求的标准有出入。根据现行的中华人民共和国国家标准 GB/T 7714-2005 文后参考文献著录规则^[1]，国内发行的论文等出版物，都应当依照此标准规定的参考文献格式进行参考文献罗列。国外的默认文献格式当然与国标要求的格式不符，那么作为患有严重强迫症

的模版作者我来说，这些问题当然一定要解决的。

4.6.1 bst 文件的结构

既然如此，就来看一下 bst 文件究竟是一个什么样的内容。bst 文件是一个纯文本文件，可以直接用任何的文本编辑器打开。最前面是一些注释，bst 文件生成时的注释，如这个模版用的 ZJUthesis.bst 文件开头的注释就说明了这个 bst 文件是根据 mbsfile_wdj_v1.0.mbs 这个文件生成了，关于 mbs 文件的作用将在后面介绍。

```
mbsfile_wdj_v1.0.mbs (with options: `seq-no,ed-au,dt-jnl,jttl-rm,pp-last,num-xser,btit-rm,
bt-rm,bkpg-x,gb-fmt,pre-edn,isbn,issn,doi,pp,xedn,no-auword,xand,nfss,')
```

开头的这一段注释说明了这个 bst 文件的生成配置为 seq-no, ed-au, dt-jnl, jttl-rm, pp-last, num-xser, btit-rm, bt-rm, bkpg-x, gb-fmt, pre-edn, isbn, issn, doi, pp, xedn, no-auword, xand, nfss, 这一长串配置选项的具体含义可以在 merlin.pdf 中查到，如 seq-no 代表的含义是文后的参考文献是按文中的引用顺序是行排序的，如果没有这个选项，就会按照作者名字的字母顺序进行排序。gb-fmt 和 no-auword 是两个针对该模版增加的两个选项，在 merlin.pdf 中是没有的，用来控制输出文献类型标志符，如期刊文章类型标志符 [J]，书籍的为 [M] 等，输出标志符在 L^AT_EX 提供的模版中，是没有这个功能的，是由我在这个模版中添加进去的。gb-fmt 是用于设置按国标 [11] 格式生成参考文献列表时采用国标格式的选项。这些设置选项是如何起作用的将在后面的内容中介绍到。

接下来就进入到 bst 文件的正题部分了，首先是一个有几十个项的列表，这个列表代表了在这个 bst 文件中用到的所有代指量名，如 author, title, year 等，这个列表可以根据 *.bib 参考文献信息库中的文档情况进行自由添加。

再接下来就是一堆的 FUNCTION 了，很长，这些 FUNCTION 就是 BibTeX 进行文献列表生成时用的程序，要控制 BibTeX 的行为，就是要修改这些 FUNCTION。这些 FUNCTION 是有层次的，在前面的是一些基本的子程序，到后面，就是由这些子程序构成的复杂功能的大程序。再到最后，就看到以 article, book, proceedings, phdthesis 等等文献类型命名的最终 FUNCTION 了，没错，就是这些 FUNCTION 直接决定了每种文献类型的输出格式。

然后，就没有然后了，这个 bst 文件就结束了。

4.6.2 BibTeX 编程的规则

前面已经讲到，控制 BibTeX 这个程序的行为就是靠 bst 文件中列出的 FUNCTION，这些 FUNCTION 的结构很像一个 C 语言文件，子程序在前，调用这些子程序的程序在后，但是这些程序的语法不是 C 语言语法，而是人读起来比较费劲的逆波兰表示法。

关于逆波兰表示法，我这儿抄一段 wiki 百科的资料列下。

逆波兰表示法 逆波兰表示法（Reverse Polish notation, RPN, 或逆波兰记法），是一种是由波兰数学家扬·武卡谢维奇 1920 年引入的数学表达式方式，在逆波兰记法中，所有操作符置于操作数的后面，因此也被称为后缀表示法。逆波兰记法不需要括号来标识操作符的优先级。

逆波兰结构由弗里德里希·鲍尔（Friedrich L. Bauer）和艾兹格·迪科斯彻在 1960 年代早期提议用于表达式求值，以利用堆栈结构和减少计算机内存访问。逆波兰记法和相应的算法由澳大利亚哲学家、计算机学家查尔斯·汉布林（Charles Hamblin）在 1960 年代中期扩充。

在 1960 和 1970 年代，逆波兰记法广泛地被用于台式计算器，因此也在普通公众（工程、商业和金融领域）中使用。

正是由于有这么一段现在在少有人使用的表达方式，使得 bst 文件中的 FUNCTION 有如天书，没有足够资料引导的话，很难读懂，也就很难调整使用。

逆波兰表示法的最大特点就是始终在操作一个栈，操作无非压栈出栈，那么这个 BibTeX 实际在操作中也是这样，把参考文献信息一个个压入堆栈，再在适当的时候释放出来，我这里以一个比较简单的 standard 也就是标准的参考文献类型的 FUNCTION 代码来解释一下这个 BibTeX 是具体如何工作的。

```
FUNCTION {standard}
{
  output.bibitem
  format.authors "author" output.check
  new.block
  format.btitle
  "title" output.check
  "[S]" *
  new.block
  address "address" bibinfo.check
  ":" *
  publisher "publisher" bibinfo.warn * output
  format.date "year" output.check
  fin.entry
}
```

我们来先看第一句 FUNCTION standard，这一句代表了 this FUNCTION 是用来处理 standard 这样一类参考文献的，如果被引用的参考文献在库中开头指明是 standard 类型，那么其相应的在 *.bbl 文件中的条目，就由这个函数来进行书写。

这个 FUNCTION 的函数体第一句 output.bibitem，这个实际是一个子函数，在 bst 文件中向前寻找，会很容易找到这个 FUNCTION output.bibitem 的定义。这里先不具体介绍该子函数的具体细节实现，其功能就是，在 *.bbl 文件中写下 “\bibitemkey” 这样一句，这里的 “key” 代表相应参考文献引用的关键字。然后在堆栈里压入一个空的字符串，相应的堆栈内容变为：

第二栈	
第一栈	一个空字符串

*.bbl 的内容如下：

\bibitem{key}

函数体第二句 format.authors，这一句是将这个参考文献中的 author 部分进行一个格式化，格式化完毕后，把这个结果压到堆栈里去，于是乎，这时候，堆栈里出现了第二个元素。

第三栈	
第二栈	作者名字
第一栈	一个空字符串

对于中文名字来说，基本用不着怎样的格式化，怎样的写法都一样，但对于拉丁字母系人名来说，可能就会有多种写法，如 John Fitzgerald Kennedy 就可能被写成 J.F.Kendey，或者 John. F. Kendey，或者 John Kendey 等等形式，具体采用哪种形式，在生成 bst 文件的时候有选项可供选择，具体选择选项参考 merlin.pdf 文档，里面有详细的说明。写到这里，不由得对拉丁语系这种极其蛋疼的姓名体系表示爱莫能助，在 bst 文档中，光这种姓名不同的处理方式就占了上百行不止的代码量，其中各种曲折分支判断，总之，无语了。。有一种情况的中文姓名需要处理，就是人名比较多的时候，在生成 bst 文件的时候，有一个选项，“nmlm “选项，用于处理显示几个人名，后面跟 x3，表示最多显示 3 个名字，跟 m3，表示显示 3 个名字还没显完的时候，加上 “et al” 这样一个后缀。另外 “xand “选项

用于去掉最后两个中文姓名之间蛋疼地用“and”这个英文词连接，避免造成的不中不洋的显示情况。

在上面一句的后面，接着了两句：“author” output.check，为什么说这是两句呢，“author”这一句表示将“author”这样一个字符串（不包括引号，下同）压到堆栈里，这时堆栈里就有两个元素了，最下面的是作者姓名，上面的一个是“author”这个字符串。

第四栈	
第三栈	"author"
第二栈	作者名字
第一栈	一个空字符串

然后output.check对刚才压入栈的两个元素进行处理，具体过程为：首先把“author”弹出来，然后去检查作者名字是不是一个空的，如果是空的，就会在 BibTeX 运行时的 log 里写出一行警告：“Warning: empty author in cite warning”。然后再将作者名字下面的空字符串栈输出到 *.bbl 文件中去，因为本来就是一个空字符串，所以 *.bbl 文件在这一步实际上并没有什么变化。

这一步做完，堆栈又成了这个样子了：

第二栈	
第一栈	作者名字

下面一句new.block，这一句是说明下面一段是新的，一般 BibTeX 输出的时候会在这一句之前的部分输出一个圆点句号。比如参考文献中，一般作者名后面跟的就是一个圆点句号。

再下面一句：format.btitle 这一句跟上面的对作者名进行格式化的作用一样，是将文献题名格式化后，比如字母变成大写之类的，压入到堆栈中去。这一句执行完成后，堆栈就变成了：

第三栈	
第二栈	文献题名
第一栈	作者名字

下面一句：“title” output.check，与名字的操作类似，也是检查一下 title 的的格式是否符合要求，然后ouput.check 这一句把**作者姓名**进行了输出，这里要注意输出的**不是文献名称**，而是前一个堆栈的内容。这一句执行后堆栈情况如下：

第二栈	
第一栈	文献题名

*.bbl 的内容如下:

```
\bibitem{key}
作者名字.
\newblock
```

下面一句"[S]"*, 其实是两个命令, "[S]" 这个命令把字符串"[S]", 压到了堆栈中, 这一句执行后的堆栈情况如下:

第三栈	
第二栈	"[S]"
第一栈	文献题名

* 这个命令的含义是将堆栈最上面两条弹出来, 把最上面那个跟下面那个合并, 再把合并的结果压栈, 这个命令的解释在文献 [9]。于是这个命令执行完后堆栈的情况如下:

第二栈	
第一栈	文献题名 [S]

下一条命令是new.block, 这个与上面一样, 也是表明文献题后将是一个点句号, 一个 block 的结束。这条命令只改变其输出状态, 在 output.nonnull 这个函数中有用。output.nonnull 这个函数后面会讲到, 这里暂时不讲这个函数。

address "address" bibinfo.check 这两条命令与前面输入作者, 题名类似, 也是将出版地格式化一下², 然后, 对这个信息元素是否存在进行一个判断。这条命令以后, 堆栈的情况变为:

第三栈	
第二栈	出版地
第一栈	文献题名 [S]

下两句":*" 与上面所说的"[S]"* 类似, 也是先把":"号压栈, 再把它与前面的出版地合起来再压栈, 这两条命令执行后, 堆栈情况变为:

²中文出版地没必要去格式化, 英文的地名可能需要格式化, 比如全大写之类的。

第三栈	
第二栈	出版地:
第一栈	文献题名 [S]

`publisher "publisher" bibinfo.warn * output` 这三句与前面的作用类似，先把出版者格式化一下，再把它与前面的“出版地:”栈合起来。

这个地方要注意一点的是：“`publisher`” `bibinfo.warn` 这两个命令是检查一下是不是不存在 `publisher` 这个元素，如果不存在，就输出一个警告。

在 `output` 命令执行前，堆栈情况为：

第三栈	
第二栈	出版地: 出版者
第一栈	文献题名 [S]

`output` 把第一栈输出，这时候要注意它输出的是文献题名信息。该条命令执行后，堆栈情况为：

第二栈	
第一栈	出版地: 出版者

`*.bbl` 的内容如下：

```
\bibitem{key}
作者名字.
\newblock 文献题名 [S].
\newblock
```

`format.date "year" output.check` 这两条命令与前面类似，将年份压入堆栈。并把前面出版地: 出版者的信息输出到 `*.bbl` 文件中去。这两条命令执行后，堆栈变为：

第二栈	
第一栈	年份

`*.bbl` 文件的内容如下：


```
\bibitem{key}
作者名字.
\newblock 文献题名 [S].
\newblock 出版地: 出版者,a
\newblock
```

^a注意这里出版者后面是逗号，不是圆点句号。

最后一条命令`fin.entry`把年份输出，堆栈清空，并在`*.bbl`内容最后面加上圆点句号结束。`*.bbl`文件的内容最终如下：`*.bbl`文件的内容如下：

```
\bibitem{key}
作者名字.
\newblock 文献题名 [S].
\newblock 出版地: 出版者,
\newblock 年份.
```

这个条目对应出来的参考文献列表格式就是现在看到的文献 [11] 列出的样子。

这样，一个最简单的标准文献类型的参考文献条目就这样被列了出来。上面讲都是一些已经成为了模块化的函数，这些函数最终都是由 37 条基本命令组合而来的，这 37 条基本命令可以在文献 [9] 中查到。

以下将以上述函数中牵涉的几个命令进行进一步的讲解。

首先看这个函数的第一句：`output.bibitem`，这是一个函数，可以在 `ZJUthesis.bst` 文件中找到它的 `FUNCTION` 定义：

```
FUNCTION {output.bibitem}
{ newline$
  "\bibitem{" write$
  cite$ write$
  "}" write$
  newline$
  ""
  before.all 'output.state :=
}
```

下面我们就来逐句解释其含义，

`newline$`，这一句含义是在 `*.bbl` 文件中新起一行。

`"\bibitem{"`，这一句是将这一个字符串压入堆栈，这一句执行过后，堆栈就变为：

第二栈	
第一栈	"\bibitem{"

write\$, 这一句的作用是将刚写入堆栈的字符串"\bibitem{" 写到 *.bbl 文件中去。

cite\$ write\$, 这两句是将引用这条文献的 key 写到堆栈中去, 然后再将这两句写到 *.bbl 文件中去。

"}" write\$, 这两句是将后面的大括号写到 *.bbl 中去。

newline\$, 再在 *.bbl 中另起一行。

""", 这一句是将一个空的内容压入堆栈, 这个地方的作用后面将会提到。

before.all 'output.state :=, 这一句有三个命令, 第一个 before.all 实际上是一个变量, 可以在 ZJUthesis.bst 文件中找到这样一句: #0 'before.all :=, 这说明 before.all 代表一个整数 0。这个地方要说明的是, 在 BibTeX 中整数的表示方式是加一个 # 号。继续回到这三个命令的解释, 先把整数 0 压入堆栈, 然后 'output.state 这一句前面的 "" 号表示后面跟的是一个变量, 表示将 output.state 这个变量名而不是变量代表的内容压入堆栈, 最后一个命令, " := " 则是将 output.state 这个变量赋予整数值 0。

再接着往下看第二句 format.authors, 这也是一个函数, 可以在 ZJUthesis.bst 中找到如下的函数定义:

```
FUNCTION {format.authors}
{ author "author" format.names
}
```

可以看出这个函数有三句: author, 这一句把 bib 文件中的相应的 author 元素提出压入堆栈, "author", 这一句把 "author" 这个字符串压入堆栈, 这个是为了后面的 format.names 这个函数输出错误信息时提示使用的。然后后面 format.names 这个就是对这个作者名字按设定好的格式进行格式化的函数, 在 ZJUthesis.bst 中也可以找到这个函数, 比较长, 这里就不再一句一句赘述了。后面在讲 *.mbs 文件时这个函数还会被提及到。

好了, 对作者名字格式化之后, 就该进行输出了, 这里有两句命令, "author" output.check。这里又出现了一个 "author", 这个是为输出时用以错误提示信息用的。output.check 这个函数是重点, 在 ZJUthesis.bst 文件中查询, 可以找到 output.check 这个函数的定义如下:

```
FUNCTION {output.check}
{ 't :=
```

```
duplicate$ empty$
{ pop$ "empty " t * " in " * cite$ * warning$ }
'output.nonnull
if$
}
```

在运行这个函数之前，堆栈情况如下：

第四栈	
第三栈	”author”
第二栈	作者姓名
第一栈	一个空字符串

第一行包含两个命令：`t:=`，第一个命令`t`代表将 `t` 这个变量名压入堆栈，在 BibTeX 中，默认定义了两个字符型变量，`s` 和 `t`，在 `ZJUthesis.bst` 文件中可以找到这两个变量的定义：`STRINGS s t`。在将 `t` 这个变量名压入堆栈后，堆栈变为：

第五栈	
第四栈	<code>t</code>
第三栈	”author”
第二栈	作者姓名
第一栈	一个空字符串

第二个命令`:=`，是将堆栈最上面两个弹出，将第二个弹出的值赋到第一个值指向的变量中去。即将`”author”` 赋给 `t` 变量，这个命令执行后，堆栈变为：

第三栈	
第二栈	作者姓名
第一栈	一个空字符串

下一行命令 `duplicate$`，这个是将堆栈最上面的一个复制一个再压入堆栈，这个命令执行后，堆栈变为：

第四栈	
第三栈	作者姓名
第二栈	作者姓名
第一栈	一个空字符串

再下一个命令 `empty$`，是判断堆栈最上面一个是不是空字符串的。具体操作过程是，先把最堆栈最上面一个弹出，判断如果是空，就将 1 压入堆栈，否则就将 0 压入堆栈。如果作者姓名不为空，堆栈就变为：

第四栈	
第三栈	0
第二栈	作者姓名
第一栈	一个空字符串

下面的一行用大括号括起来了：`{ pop$ "empty" t * " in " * cite$ * warning$ }`。用大括号括起来代表把这一串代码作为一个嵌入的子函数，然后把这个子函数的入口压入堆栈。如果执行这个子函数，就是如下一串操作：把堆栈最上面一个元素弹出，输出一个字符串，比如：“`empty author in citekey`”，看到了吧，这个地方的 `t` 就变成了“`author`”，这是前面几句进行的赋值。

第五栈	
第四栈	嵌入子函数的入口
第三栈	0
第二栈	作者姓名
第一栈	一个空字符串

下面一行命令`'output.nonnull`，这个前面的`'`号还是表示把这个函数的入口压入堆栈，当然也可以用`{output.nonnull}`。

这一句执行后，堆栈情况如下：

第六栈	
第五栈	<code>output.nonnull</code> 函数的入口
第四栈	嵌入子函数的入口
第三栈	0
第二栈	作者姓名
第一栈	一个空字符串

最后一个命令 `if$` 一个比较关键的命令，也是一个常用的命令。这个命令的操作是将堆栈最上面三个元素弹出，然后，根据第三个弹出的元素是 0 还是 1 进行判断，如果是 0，就执行第一个弹出的元素指向的函数，如果是 1，就执行第二个弹出的元素指向的函数。这实际上就是一个分支判断函数。我们来分别来看一下第三栈里是 0 和是 1 两种情况的执行结果吧。

弹出三个元素后，堆栈就变为：

第三栈	
第二栈	作者姓名
第一栈	一个空字符串

1. 如果第三个弹出的元素是 1，那么就说明作者姓名是个空的字符串，那么就执行那一长串命令组成的函数：`pop$ "empty" t * "in" * cite$ * warning$`。第一个命令，`pop$`，把空的作者姓名弹出。堆栈变为：

第三栈	
第一栈	一个空字符串

后面的一串就是往堆栈中压入元素，然后再合并，压入，再合并，到 `warning$` 命令之前，堆栈变为：

第三栈	
第二栈	"empty author in citekey"
第一栈	一个空字符串

这里面这个“`citekey`”指给相应参考文献的别名。可以看到，这个时候堆栈最上面写了一条警告信息，那么下面紧跟的命令 `warning$` 就把这条警告信息输出到 log 文件和输出命令行里去了。

2. 如果第三个弹出的元素是 0，则表示作者姓名不是空字符串，这是正常情况，这个时候就执行 `output.nonnull` 这个函数，把作者姓名前面的信息段写入到 `*.bbl` 文件中。这个地方一定要注意，这时写入的并不是作者姓名这个信息，而是它下面的第一栈，一个空字符串。为什么是这样，下面将继续介绍 `output.nonnull` 这个函数在做什么。

下面接着向下介绍这个 `output.nonnull` 函数。在执行这个函数之前，堆栈情况如下：

第三栈	
第二栈	作者姓名
第一栈	一个空字符串

`output.nonnull` 函数定义如下：

```
FUNCTION {output.nonnull}
{ 's :=
  output.state mid.sentence =
    { ", " * write$ }
  { output.state after.block =
    { add.period$ write$
      newline$
      "\newblock " write$
    }
    { output.state before.all =
      'write$
      { add.period$ " " * write$ }
      if$
    }
    if$
  mid.sentence 'output.state :=
}
if$
s
}
```

第一行's :=，这是两个命令，第一个命令，把 s 这个变量名放进堆栈，第二个命令:=，把 s 变量名弹出，再把作者姓名这个字符串弹出，把作者姓名这个字符串赋给变量 s。这两个命令完成后，堆栈变为：

第二栈	
第一栈	一个空字符串

下面的命令实际是几个嵌套的判断分支，最外层的分支是这样一个判断：

```
output.state mid.sentence =,
```

这一行共三个命令，前两个都是变量，一个是 output.state，另一个是 mid.sentence，从前面已经知道，output.state 已经被赋值为 before.all，即一个整数值 0，而 mid.sentence 的值是 1，第三个命令 = 号就是判断一下前面两个数值是不是相等的，结果当然是不等，于是一个 0 就压进了堆栈之中，这样，程序就进入了下面的分支：

```
output.state after.block =
```

这一句与上面一句类似，也是进行一个判断，明显，output.state 也不与 after.block（值为 3）相等，于是又进入了下面一个分支：

```
output.state before.all =
```

这次相等了，于是运行紧挨着它的分支：这个分支只有一个命令：write\$，这个命令就是将堆栈中最上面的一个元素写到 *.bbl 文件中去，根据此时堆栈的状态，此时写入 *.bbl 的只是一个空字符串，也可以说什么也没有写。*.bbl 的内容仍是：

\bibitem{key}

执行之后的下一句命令

`mid.sentence`output.state :=`

把 `output.state` 的值改为 `mid.sentence`（其值为 1）

最后一行的一句 `s`，把刚才存进去的作者姓名，又压回了堆栈，这条命令之后，堆栈内容变为：

第二栈	
第一栈	作者姓名

下面让我们回到 `standard` 这个类的函数中来，下一句函数是 `new.block`，我们找到它的函数定义如下：

```
FUNCTION {new.block}
{ output.state before.all =
  'skip$
  { after.block 'output.state := }
  if$
}
```

这个函数的操作过程如下，先对 `output.state` 进行判断，根据上面提到的，现在的 `output.state` 的值是 `mid.sentence`，这里就与 `before.all` 不相等了。于是执行 `after.block`output.state :=` 这一行，把 `output.state` 的值又改为 `after.block`。

接下来的两行命令与前面的比较类似，

`format.btitle " title" output.check`

对文献标题进行格式化，然后再进行输出，但与前面输出不同的时，此时 `output.state` 的状态是 `after.block`。该状态在 `output.nonnull` 中对应的运行行是：

```
add.period$ write$
newline$
"\newblock " write$
```

运行该段命令段前，堆栈的状态是：

第二栈	
第一栈	作者姓名

这一段命令行解释如下：`add.period$` 是在当前堆栈最上面一个字符串元素后面加一个“.”号，就是英文句号。这条命令后，堆栈的状态变为：

第二栈	
第一栈	作者姓名.

注意“作者姓名”后面紧跟的英文圆点句号。

下一条命令是 `write$`，就是把堆栈中最上面的元素写到 `*.bbl` 中，这条命令之后，`*.bbl` 的内容变为：

```
\bibitem{key}
作者名字.
```

再下面两行命令

`newline$`

`”\newblock” write$”`

则是在 `*.bbl` 中重启一行，并写下了`”\newblock”` 这样一个字符串，这两行命令执行后，`*.bbl` 文件内容如下：

```
\bibitem{key}
作者名字.
\newblock
```

这一段命令执行后，`output.state` 再次被修改为 `mid.sentence`，并且文献题目被从 `s` 变量中再次压回堆栈。堆栈变为：

第二栈	
第一栈	文献题名

其后的函数与上面所述函数作用、结构相同，在后面不再详述。

4.6.3 bst 文件的调试技巧

从上面的 `bst` 文件的编程例子可以看出，`BibTeX` 就是按照操作堆栈以及输出堆栈中字符串的方式，实现对 `*.bbl` 文件的自动编写的。

那么，在掌握了 *.bst 文件的编程规则之后，就可以自己编写自己需要的参考文献格式，那么在编写过程中，自然要不断调试输出的效果以及查找问题所在，这个时候就需要一些 debug 的方式来辅助编程。这一小节将对这些编程中常用的调试方式进行一下介绍。

在调试 *.bst 文件的输出效果时，当然不用每次都用一篇文章来一次次试验其输出效果，大部分时候，只需要运行一次 BibTeX 这个程序，再看一下 *.bbl 中的内容，即可判定该 *.bst 文件是否符合要求。

如果 *.bbl 文件中输出的内容不对，那么就需要查看 BibTeX 运行中的堆栈情况，查看堆栈我推荐以下几个命令组合进行检查。

1. duplicate\$ top\$

这两个命令的操作分别是复制堆栈最上面一个元素，把最上面一个元素弹出并输出其值到 BibTeX 运行输出命令行中去。这样，这两个命令的组合实际上就成了运行过程中查看堆栈最上面一个元素的值，起到了 debug 中查看当前程序内存状态的作用。

2. swap\$ duplicate\$ top\$ swap\$

这个与上面类似，不同的是这个命令用来查看堆栈最上面第二个元素的值。一般情况下，运行中可能牵涉两个元素的值，用这个方法与上面的方法结合，就可以实现查看堆栈最上面两个元素值的作用。

3. stack\$

这个命令是把堆栈中所有元素一股脑儿全倒出来并显示出来，一般这个调试方法不太常用。

4. top\$

把需要查看的状态量，比如 output.state，先压入堆栈，再用 top\$ 这个命令查看它的值。注意这里 top\$ 与 pop\$ 命令的区别，前者是将元素弹出后再显示出来，后者是只弹出，不显示。

一般情况下，对 *.bst 文件进行调试使用上面几种调试命令就可以了，将这些命令插入到 *.bst 文件中的合适位置，就可以查看 *.bst 文件在运行中的相应状态。调试完后，不要忘记将它们从 *.bst 文件删除。

4.7 *.mbs 与 *.dbj 文件

前面已经讲到, BibTeX 这个程序工作设置主要就是参考 *.bst 文件, 有什么样的 *.bst 文件, 就有什么样的参考文献格式输出。从而实现了参考文献的自动格式化输出, 当需要改变参考文献的输出格式时, 只需要更换相应的 bst 文件即可。

但是, 从上面一节介绍 bst 文件的语法规则及使用技巧的情况来看, bst 并不是一个很好用的格式, 它的语法太过晦涩, 即便是对其语法特点有较好的理解, 理清各个子函数之间的关联脉络也是一件比较费心思的事。为了解决这个问题, 又出现了更高一级的 bst 文件配置方案: mbs 文件和 dbj 文件就是这样一种配置方法。

这里我先不谈 mbs 文件和 dbj 文件的内容结构, 先讲一下这种配置方法的基本思路。既然 bst 文件是那样晦涩难懂难编写, 那么作为开发者, 就先编写一套涵盖大部分人应用的, 比较完整的 bst 格式输出集, 然后再定一些相对较为简单的选项配置文件, 比如人名要显示几个, 文献来源要不要粗体或者斜体处理之类的, 利用这些配置选项, 把所需要的 bst 语句抽出来, 写到 *.bst 文件中, 这样就生成了我所需要的参考文献格式。这种思路实际上就类似于人一般做事的思路, 比如一个饭店的厨师, 顾客说要少放盐, 我就放盐的时候, 少放一点点, 多放辣椒, 就放的时候多放一些, 先煮后炸, 或者先炸后煮, 同一道菜, 不同顾客不同要求, 在流程中作一些小的改动, 就能出很多种花样儿来。

4.7.1 一般的 bst 文件的生成过程

根据参考文献 [8]makebst.pdf 中的说明, 一般情况下, 只需要运行

```
tex makebst.tex
```

然后就会有一系列问答式的选项进行回答, 如生成的 dbj 文件和 bst 文件的文件名, 这里我用的是 dbjfile_wdj_1.0 和 ZJUthesis 其它如显示几个作者名, 文献标题是否采用粗体或者斜体等等问题, 回答完之后, 就可以得到一个名为 “dbjfile_wdj_1.0.dbj” 的文件。

然后再运行

```
tex dbjfile_wdj_1.0.dbj
```

就可以得到一个 ZJUthesis.bst 的文件, 这个就是一个 *.bst 文件。当然这个文件是用 L^AT_EX 默认的选项中得来的, 前面就已经提到, 这个默认的选项并不适合论文中对参考文献格式的要求。那么要得到符合论文参考文献格式要求的 bst 文件, 要么直接对 bst 文件进行修改, 要么就修改这个 bst 文件的源头 mbs 文件及相应的配置文件 dbj。下面将对 mbs 文件和 dbj 文件进行一个介绍及配置说明。

4.7.2 有所不同的注释方式

使用 \LaTeX 中已经习惯，以 “%” 起头的部分就是注释，不论是在每行的起始还是行中。但在这里，在 *.mbs 文件中，这个规则要被打破一下。

但这并不是说在 *.mbs 文件中，以 “%” 起头的部分就不是注释了，而是说，有一部分，不是注释。是哪一部分呢，就是在每一行的开头是 “%”，然后后面又紧跟一个 “<”，这种情况，“%” 就不代表后面跟的是注释，而代表是一个选项了。

比如在 merlin.mbs 文件中，可以找到很多这种行，比如：

```
%<ay> author format.key output
```

这一句就是典型的非注释选项语句，该句的意思是，如果 “ay” 这个选项被使用，那么后面跟着的这一行命令：author format.key output，就会被写到生成的 *.bst 文件中去。

除了这样一种选项设置方式外，还有以下几种选项设置方式：

- %<!ay> author format.key output

这种方式的意思是，如果 “ay” 这个选项没有被使用，那么后面跟着的这一行命令，就会被写到生成的 *.bst 文件中去。

- %<*ay>

```
author format.key output
```

```
%</ay>
```

这种方式的可以用来对多行命令进行设置，即 %<*ay> 与 %</ay> 之间的行，在选项 “ay” 被使用时，写入到生成的 *.bst 文件中去。

- %<*&!ay>

```
author format.key output
```

```
%</!ay>
```

这种方式与上面的类似，也是在选项 “ay” 未被使用时，将多行命令写入到 *.bst 文件中去。

- %<ay&alph|!au-col> author format.key output

除了上述所说的方式外，多个选项还可以用 “!”，“&”，“|” 这些逻辑运算符连接起来完成多个功能。

关于上述这些选项的设置方式，在参考文献 [8] 中有详细说明。 \LaTeX 就是根据这种设置方法，根据 dbj 文件中给出的选项，从而将符合选项的语句从 *.mbs 文件中抽取出来，形成一个新的 bst 文件。

4.7.3 集大成的 *.mbs 文件

*.mbs 文件实际上就是一个包含了大部分可能设置情况的文件， \LaTeX 生成 bst 文件实际就是从 *.mbs 这个库中取出了所需的部分。在 *.mbs 文件中，除了有这些说到的包含大部分情况的 bst 语句，还有各种选项的定义及注释，甚至整个 merlin 文件的使用说明书都包含在里面。除了刚才说到的 bst 语句，下面看一个选项的定义及注释。

```
\beginoptiongroup{STYLE OF CITATIONS:}{ }
\optdef{*}{ }{Numerical}{as in standard LaTeX}
\optdef{a}{ay}{Author-year}{with some non-standard interface}
\optdef{b}{alph}{Alpha style, Jon90 or JWB90}{for single or multiple authors}
\optdef{o}{alph,alf-1}{Alpha style, Jon90}{even for multiple authors}
\optdef{f}{alph,alf-f}{Alpha style, Jones90}{(full name of first author)}
\optdef{c}{cite}{Cite key}{(special for listing contents of bib file)}
\getans
\endoptiongroup
```

`\beginoptiongroup{STYLE OF CITATIONS:}{ }` 这一句表示开始一组设置，这一组设置中的选项只能选择一个。后面的大括号里内容“STYLE OF CITATIONS”表示这个设置是做什么用的，起到注释的作用。

`\optdef{*}{ }{Numerical}{as in standard LaTeX}` 这一句表示这一组设置的默认设置选项，空的大括号表示默认不需要选项。再后面的大括号中内容“Numerical”表示这是一个使用示例，这个示例其实不明显，后面的几个选项中“Alpha style, Jon90”就是一个明显的例子。再后面的大括号中内容“as in standard LaTeX”是这一个选项的说明注释。

`\optdef{a}{ay}{Author-year}{with some non-standard interface}` 这一句代表在执行 dbj 文件生成时，选择 a 就选择这个选项的意思，其实后面的内容与默认选项作用相同，分别是选项“ay”，例子“Author-year”和“with some non-standard interface”。

最后两句:

```
\getans
\endoptiongroup
```

是这个选项的结束语句，这些语句的具体含义可参考文献 [7]。

除此之外，mbs 文件中还有关于此 mbs 文件的使用说明，只是都进行了注释。

mbs 文件中关于各种 bst 设置情况的判断比较多, 很多函数都有多个条件判断, 使用哪一句命令。整个下来使得 mbs 文件后面关于 bst 源码的部分看起来是一团乱麻, 在看及更改这部分代码的时候要时刻注意每行代码的有效条件。

针对参考文献 [11] 中对参考文献格式的要求, 本文对 mbs 文件增加了三个选项: rtf-n, gb-fmt, no-auword, 这三个项的定义如下:

```
\beginoptiongroup{REFERENCE TYPE FLAG:}{}
\optdef{*}{}{The reference type flag exist}{}
\optdef{n}{rtf-n}{The reference type flag not exist}{}
\getans
\endoptiongroup

\beginoptiongroup{YEAR COLON PAGES FLAG:}{}
\optdef{*}{}{Use normal format}{2000, 12-23}
\optdef{g}{gb-fmt}{Use GB/T7714 format}{2000:12-23}
\getans
\endoptiongroup

\beginoptiongroup{SEPARATE OUT FLAG:}{}
\optdef{*}{no-auword}{The separate out reference type flag exist}{//editorname, BookTitle}
\optdef{n}{}{The separate out reference type flag not exist}{In editorname, editor, BookTitle}
\getans
\endoptiongroup
```

rtf-n 选项用来设置参考文献标题后不放置参考文献标志符, 如期刊文章用 “[J]”, 书籍用 “[M]” 等, 所以对于本论文的参考文献格式要求, 这个选项不使用。

gb-fmt 用来实现 GB/T 7714-2005 中要求的“年份: 页码”这种格式, 这种格式在 L^AT_EX 自带的 merlin.mbs 中不提供这种支持, 所以这里我自己做了一个这样的选择并把它放进了我修改的 mbs 文件中。有兴趣的可以在这个文件中搜索这个选择的使用处, 从而了解这个选项的配置方法。

no-auword 同样是为了满足 GB/T 7714-2005 中关于析出文献的格式, 让析出文献与所在文献集之间存在双斜线表示析出关系, 并能分别给出作者及编者的姓名。这个选项的使用同样在我修改的 mbs 文件中搜索可以找到, 从而了解对这个功能的实现方法。

4.7.4 *.dbj 选项文件

dbj 选项文件其实是比较简单的, 打开这个文件就可以看到, 它实际是一个选择说明及选项选择的一个列表文件, 通过选择相应的配置选项, 就可以生成得到一个符合要求的 bst 文件。由于前面我在 mbs 中增加了选项, 因此这个 dbj 文件中同样增加了相应的选项呼应。注意要对我的 bst 文件进行修改, 最后使用我提供的配套的 mbs 文件与 dbj 文件。

关于 dbj 文件中各个选项的具体作用, 参考文献 [7], 这些选项的已经相对完全了, 但

对于世界上这么多种格式要求，还是不能完全满足的，至少国内的需求是满足不了。而且国外对于参考文献格式的思路与国内还是有一种说不出的不同。所以，如何做出一个适合国内大部分编辑部及院校要求的参考文献格式大模版，还是一件很有挑战性的工作，期待各位读完我这一部分内容的各位 TeXer 的一起努力。

第 5 章 一些反馈的问题

以下将对一些在这个模版发布两年间，使用的各位网友发给我邮件询问相关问题的内容的解答作一个集中记录。其中的一些问题及我提供的解决方案可以供广大有不同需要的网友们参考。

5.1 关于使用 author year 参考文献引用方式的问题

这是我收到的第一封回复邮件，问题相对比较简单，是一个叫 Jerry Chen 的网友给我的，这个模版默认是使用数字编号对标示参考文献引用的，Jerry Chen 网友希望使用 author year 这种格式进行参考文献的标注。就像如下的格式。

`\citet{jon90}` \rightarrow Jones et al. (1990)

要实现这个显示效果，需要对 ZJUthesis.cls 文件以及 ZJUthesis.bst 文件进行修改，其中对 ZJUthesis.bst 文件修改可以通过修改 dbjfile_wdj_V1.0.dbj 文件的方式进行修改。分别如下：

- ZJUthesis.cls 文件中

第 41 行是关于正文中参考文献引用标记格式设置的。把它由数字排序方式变为作者年代的排序方式，即由：

```
\RequirePackage[sort&compress,longnamesfirst,square,super]{natbib}
```

修改为：

```
\RequirePackage[longnamesfirst,round,authoryear]{natbib}
```

第 654 行是关于正文后参考文献列表的结构格式，将其由数字标号方式改为作者年代标记方式。即由：

```
\setcitestyle{numbers, round, comma, aysep={}, yysep={}, notesep={},}
```

修改为：

```
\setcitestyle{authoryear, round, comma, aysep={}, yysep={}, notesep={},}
```

- dbjfile_wdj_v1.0.dbj 文件中

对选项 %STYLE OF CITATIONS: 修改为: ay

对选项 %MAX AUTHORS BEFORE ET AL: (if regular cite not selected) 修改为: mct-1, %:
One et al

然后再生成新的 ZJUthesis.bst 文件即可, 该文件的生成方式见第五章内容中介绍。

5.2 关于 chapter 居中格式的问题

这个模版中, 每一章的标题默认是左对齐设置的, 当然有的同学想设置成居中, 比如给我发邮件的 dongliang 同学。这个也很简单, 只要修改 ZJUthesis.cls 文件中的第 546 行, 由:

```
\CTEXsetup[format={\noindent}]{chapter}
```

修改为:

```
\CTEXsetup[format={\centering}]{chapter}
```

即可, 关于该处设置的含义可以参考 CTeX 自带的帮助文档 ctex.pdf。

5.3 关于章级目录有时居中有时不居中的解决方案

这个问题有点儿类似上面的情况, 要求更复杂一些, 是由叫 zwb 的网友提给我的。但这个问题的解决方案更简单, 只要在需要居中的章节前加上

```
\CTEXsetup[format={\centering}]{chapter}
```

在需要左对齐的章名前加上

```
\CTEXsetup[format={\noindent}]{chapter}
```

即可。

5.4 关于标题两行还写不下的问题

这是一个叫 FRW 的同学提给我的, Ta 的标题太长, 我的模版里只设置了两行写标题, 需要第三行, 这就需要修改 ZJUthesis.cls 文件来适应这个问题了。其实跟添加第二行的方式一样, 只是增加了一个第三行内容的命令及与第二行相同的判断。

首先增加两个命令 `\EtitleB` 和 `\englishtitleB`，再对这两个命令的使用位置进行定义。

这两个命令的定义语句如下：

```
\newcommand\EtitleB[1]{\def\ZJU@value\EtitleB{#1}}
\newcommand\englishtitleB[1]{\def\ZJU@value\englishtitleB{#1}}
```

在两处对标题多行判断的后面加上这样几句：

- 在首页上的题目部分

```
\fi\\[3mm]
% 第三行英文标题
&
\ifx\ZJU@value\EtitleB\undefined
\hfil
\else
{\bfseries\zihao{-2}\ZJUunderline[260pt]{\ZJU@value\EtitleB}}
\fi\\
```

第一行的 `\fi\\[3mm]` 意思是从这个 `\\fi\\` 处后面开始加代码，这个 `3mm` 是为了每一行高度都一样设置，这个从上面第一行最后一句就可以看出来。增加代码中的“&”符号是因为这个地方用的是 `tabular` 环境用于对齐。

- 在英文标题页的部分

```
% 判断英文标题有无第三行
\ifx\ZJU@value\englishtitleB\undefined
\hfil
\else
\ZJUunderline[300pt]{\ZJU@value\englishtitleB}
\fi}
```

增加的代码与上面一条类似，不再多述。

5.5 目录层次与子目录分层缩进

FRW 同学还提出了另一个问题，这个模版的目录中只有两层标题，想要三层标题，而且这个模版中目录两层标题的字体字号都一样，想要不同层次有不同缩进。这个问题也容易解决，都在 `ZJUthesis.cls` 中有相应命令设置。第 601 至第 620 行是关于目录的格式设置，比如增加及修改下面的所列，就可以满足上面的要求。

```

\renewcommand{\cftsecpagefont}{\rm\zihao{-4}}
\renewcommand{\cftsubsecleader}{\cftdotfill{\cftdot}}
\renewcommand{\cftsubsecfont}{\fangsong\zihao{-4}}
\renewcommand{\cftsubsecdotsep}{\cftdotsep}
\renewcommand{\cftsubsecpagefont}{\rm\zihao{-4}}
\setlength{\cftbeforechapskip}{-2pt}
\setlength{\cftbeforesecskip}{-2pt}
\setlength{\cftbeforesubsecskip}{-2pt}
\setlength{\cftsecindent}{2em}
\setlength{\cftsubsecindent}{4em}
\setcounter{tocdepth}{2}

```

这几句增加了 subsection 一级的目录显示格式，把 section 及 subsection 目录列表前面的缩进设置为 2 个字符和 4 个字符，最后又把目录的显示深度由原来的 1 设置为 2，就可以显示三级标题了。

5.6 关于分章参考文献的用法

这个模版里头的参考文献是一个章节格式的，全文只有一个参考文献章，这是一个一般的情况。当然有的同学希望能采用每一章都有自己参考文献的解决方案。这个方案也比较简单，只用修改如下几个地方。

1. 使用 chapterbib 包

首先在导言区，加入 chapterbib 包，带上 sectionbib 选项。

2. 每章增加参考文献命令

这个模版的源文件每一章都是一个甚至多个独立的 tex 文件，并在主文件“论文模版示例.tex”中用“include”命令¹将其包含在主文件中。要在每章的 tex 文件的最后，加上 \ZJUthesisbib{thesisbib} 这一条命令，假如是把所有章的参考文献数据库都写在一个文件里，比如这个模版中的 thesisbib.bib，那这个命令的参数在所有章中都是“thesisbib”。如果每章的参考文献数据库都有各自的独立的数据库文件，那么每章中这个命令的参数就不同。

3. 删去原来的参考文献引用命令

把全篇最后的参考文献引用命令删去，用不到了。

¹这个地方要注意不要使用“input”命令，使用这个命令不能实现分章参考文献

4. 修改编译命令

原来生成 PDF 文件时，bibtex 运行是一条命令“bibtex 论文模版示例”，现在就要根据有几章有参考文献列几条不同的 bibtex 命令了。即：

```
bibtex .\Chapter\chap1
bibtex .\Chapter\chap2
bibtex .\Chapter\chap3
bibtex .\Chapter\chap4
bibtex .\Chapter\chap5
bibtex .\Chapter\chap6
```

5.7 第 X 章格式的修改

这个模版的每一章的章节号直接是阿拉伯数字，有同学想用第 X 章这种格式，修改也很简单，根据 CTeX 自带的帮助文档 ctex.pdf，只要将 ZJUthesis.cls 中的第 492 行由

```
\CTEXsetup[name={,}]{chapter}
```

修改为：

```
\CTEXsetup[name={第,章}]{chapter}
```

即可。至于字体修改，居中还是偏左，都是在这几行里进行修改，具体命令参数意义参考 ctex.pdf。

5.8 多个参考文献文中标格式

如果在正文中某处引用多个参考文献，且是用数字序号进行标注，那么就牵涉一个数字标号间标点符号以及连续数字序号的缩写问题。这些设置都在 natbib 包引用时候的参数中。本论文模版的 natbib 包引用在 ZJUthesis.cls 文件的第 37-41 行，有关代码如下：

```
% sort&compress参数用于按引用顺序排列参考文献
% longnamesfirst参数用于处理长人名顺序，将first name排前面，用于外国人名
% square参数，引用标号用方括号括起
% super参数，引用标号为上标格式
\RequirePackage[sort&compress,longnamesfirst,square,super]{natbib}
```

chenchao 同学曾发邮件问我如何实现类如 [1-6] 这种文献引用标注的, 这个实现就是靠 `sort&compress` 这个参数。同时, 这些设置最后的形如 [2;4;7-9] 这种引用标注中用的是分号, 如果要改用逗号, 只要在参数中增加一个 `comma` 参考即可, 即:

```
\RequirePackage[sort&compress,longnamesfirst,square,super,comma]{natbib}
```

5.9 关于每一章标题头上的空白部分

有的同学觉得这个模版每一章的标题前到页眉的空白太大, 想作调整, 这个地方的调整也是参考 CTeX 的帮助文件 `ctex.pdf`, 其中关于 `beforeskip` 和 `afterskip` 部分的设置方式, 将其设置小一些即可。

此外, 还有同学问我如何让每一章标题那页上也有页眉, 这个问题也比较简单, 只要把 `ZJUthesis.cls` 第 517 行至 527 行对 `plain` 类型页眉页脚的定义改成与下面紧接着的 `fancy` 类型一样就可以了。不过这里我并不建议这样做, 因为每章的第一页还是不加页眉比较好看一些。

5.10 GBK 与 UTF 版本的问题

有的同学希望用 UTF-8 版本, 这个版本现在已经解决了 GBK 与 UTF-8 版本兼容的问题, 这个模版同时发布两个版本, 分别为 GBK 版与 UTF-8 版, 给不同需要的同学使用, 两个版本生成的文档除英文字体略有不同外, 其余格式是完全相同的, 且两个版可以互相直接转换。

GBK 版与 UTF-8 版的唯一一点区别在一个字体包的引用。在 UTF-8 版中, 使用的是 `fontspec` 包, 在 GBK 版中, 使用的是 `times` 包。这两个包的引用在 `ZJUthesis.cls` 最前面可以找到。

本 github 项目只包含 utf8 版本, 如需要 GBK 版本请参见原 google code 项目:
<http://code.google.com/p/zjuthesisistex/downloads/list>。

第 6 章 其他一些使用技巧

如果在生成文档时发生错误，不要惊慌，可以先把生成的文件全部删除再试一次。就是把除了 `tex` 文件外的其它同名文件都删掉。

使用 WinEdt 编辑 `tex` 文件时，如果嫌命令太长打着费劲，试试只输前几个字母然后按“`Ctrl+Enter`”键，哈！WinEdt 替你把剩下的部分补全了。

遇到问题不要慌，看下方小窗口里提示的出错信息，会有很多提示你错在哪里的。

不同系统下生成的 `eps` 可能会有兼容问题，如本模版中的 `setroot.eps` 和 `rffndb.eps`，在 `xp` 和 `windows 7 x64` 似乎不能通用。解决方案很简单，只要用 `bmeps -p 1 -c setroot.jpg setrooteps` 重新生成一次即可解决。`rffndb.eps` 生成命令同 `setroot.eps`。

这个文档我用的是 `gVim` 编辑的，`gVim` 自带的自动补全功能比 WinEdt 更强大，让我在编写这个文档时省了不少重复工作量。

如果会使用 `make` 程序，那么使用 `Makefile` 来生成文档更方便一些。

在 UTF-8 版本中，如果一个命令后紧跟汉字，比如像这样“`songti`好的”，编译的时候就会报错，处理办法就是在命令后面加一个空格或者一个大括号，就像这样：“`songti 好的`”或者“`songti{}好的`”

差不多了，就写这几条吧，想起来什么再写。

把另外几个参考文献当引用例子使用一下：专利^[12]，标准^[13]，电子文档^[14]，期刊文章^[15]，学位论文^[16;17]。

这份文档从规划到完成，历时近 20 日，也是自己 `LaTeX` 学习一个总结吧。

参考文献

- [1] CTeX 论坛. 一份不太简短的 \LaTeX 2 ϵ 介绍 [EB/OL]. 中国: 中文 \TeX 学会, 2007. <http://mirrors.ustc.edu.cn/CTAN/tex-archive/info/lshort>.
- [2] Knuth Donald Ervin, Bibby Duane Robert. The \TeX book[M]. United States of America: Addison-Wesley Publishing Company, 1996.
- [3] 张林波. 关于新版 CCT 的说明 [EB/OL]. 中国: 中国科学院数学与系统科学研究院, 2006. <ftp://ftp.cc.ac.cn/pub/cct/README.pdf>.
- [4] 王磊. \LaTeX 2 ϵ 插图指南 [EB/OL]. 2000. <http://mirrors.ustc.edu.cn/CTAN/info/epslatex/english/epslatex.pdf>.
- [5] Lapo Filippio Mori. Table in \LaTeX 2 ϵ : Packages and methods[J]. The \PracTeX Journal, 2007. (1):1–38.
- [6] Patrick W. Daly. Natural sciences citations and references[EB]. 2009. <doc/texlive-latex-base-doc/latex/natbib.pdf>.
- [7] Patrick W. Daly. A master bibliographic style file[EB]. 20011. <doc/latex/custom-bib/merlin.pdf>.
- [8] Patrick W. Daly. Customizing bibliographic style files[EB]. 2003. <doc/latex/custom-bib/makebst.pdf>.
- [9] Oren Patashnik. Designing bibtexing style[EB]. 1988. <doc/texlive-base/bibtex/base/btxhak.pdf>.
- [10] Oren Patashnik. Bibtexing[EB]. 1988. <doc/texlive-base/bibtex/base/btxdoc.pdf>.
- [11] 全国信息与文献标准化技术委员会. GB/T7714-2005 文后参考文献著录规则 [S]. 北京: 中国国家标准化管理委员会, 2005.
- [12] 王东举. \LaTeX ZJU 学位论文模版 [技术型].
- [13] ZJUEE. \LaTeX 2 ϵ 使用标准说明 [S]. 杭州: 鼠尾出版社, 2010.
- [14] 张林波. CCT DOS 版参考手册 [EB/OL]. 北京: 中科院计算数学与科学工程计算研究所, 1997. <http://www.url.com>.
- [15] 罗振, 田丰, 孙小平, 孙恩岩. 基于 LATEX 的学位论文模板的设计与实现 [J]. 沈阳航空工业学院学报, 2007. 24(3):45–48.
- [16] 作者. 我的测试论文题目 [D]. 杭州: 浙江大学, 2008.
- [17] 作者. 我的测试硕士论文题目 [D]. 杭州: 浙江大学, 2008.

附录 A - 贡献者

shuwei1204@163.com: 主要贡献者，原始项目地址：
<http://code.google.com/p/zjuthesistex/downloads/list>

ibillxia@gmail.com: 当前项目创建者，在原始项目上做了少许修改和扩充。

附录 B - 版本更新

版本更新记录。

索引

L^AT_EX, [II](#), [III](#)

T_EX, [VI](#), [3](#)

Acrobat, [26](#)

bmeps, [26](#)

booktabs-de, [32](#)

CCT, [II](#)

CJK, [II](#)

C_TTeX, [II](#)

custom-bib, [35](#)

dvipdfm, [III](#)

dvips, [II](#)

eps, [9](#), [26](#)

equation, [35](#)

graphics, [29](#)

graphicx, [27](#)

hyperref, [III](#)

jpg, [26](#)

MikTeX, [II](#)

multirow, [33](#)

natbib, [35](#)

pdfTeX, [II](#), [26](#)

photoshop, [26](#)

slashbox, [34](#)

twoside, [10](#)

Word, [1](#)

x64, [II](#)

公式编辑器, [1](#)

勘误, [I](#)

图片, [26](#)

扩展包, [27](#)

矢量, [27](#)

缩放, [27](#)

表格, [26](#)

作者简历

1. 第一条的内容
2. 第二条内容

发表文章目录

- 1. 第一篇
- 2. 第二篇

致 谢

在我写这个文档的过程中, 得到了网络上很多网贴的帮助, 在此感谢 baidu, Google, 感谢 CTeX 社区 <http://www.ctex.org>, L^AT_EX 学习园地: <http://blog.sina.com.cn/wangzhaoli11>, 中科大 CTAN 镜像 <http://mirrors.ustc.edu.cn/CTAN/>, 水木社区 T_EX 版等网站、论坛, 其他一些较小的个人网站, 论坛不再一一点名, 在此一并感谢。感谢浙江大学数学系提供的原始模版, 感谢 88T_EX 版。